HOCHSCHULE
MITTWEIDA
UNIVERSITY OF
APPLIED SCIENCES

BACHELORARBEIT

Herr
Robin Becker

Entwurf einer Antriebs-schlupfregelung für einen Formula Student Rennwagen

Mittweida, 2012

Fakultät Maschinenbau

BACHELORARBEIT

Entwurf einer Antriebs-schlupfregelung für einen Formula Student Rennwagen

Autor:
Herr

Robin Becker

Studiengang:
Mechatronik

Seminargruppe:
ME09w1-B

Erstprüfer:
Prof. Dr.-Ing. Martin Zimmermann

Zweitprüfer:
Prof. Dr.-Ing. Hans-Gerhard Kretzschmar

Einreichung:
Mittweida, 12.09.2012

Verteidigung/Bewertung:
Mittweida, 2012

Bibliografische Angaben:

Becker, Robin
Entwurf einer Antriebsschlupfregelung für einen Formula Student Rennwagen - 2012 – 53 Seiten, 24 Abbildungen, 6 Tabellen,
Hochschule Mittweida (FH), University of Applied Sciences,
Fakultät Maschinenbau, Bachelorarbeit 2012

Inhalt

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

TMM	Technikum Mittweida Motorsport
ASR	Antriebsschlupfregelung
SAE	Society of Automotive Engineers
EIT	Elektro- und Informationstechnik

1 Übersicht

Im einleitenden Kapitel werden die Motivation und die Zielsetzung der Bachelorarbeit dargelegt.

1.1 Motivation

Begründet als eine Fortsetzung zu einem bereits im Komplexpraktikum Mechatronik abgehandelten Thema wird auch diese Bachelorarbeit als ein Projekt für das Formula Student Team der Hochschule Mittweida angesehen. Dieses während des Komplexpraktikums entstandene Projekt befasste sich mit dem Entwurf einer elektropneumatischen Schaltung für einen Formula Student Rennwagen.

Die Veranstaltung, welche im Hintergrund der Projekte steht, heißt Formula Student. Diese existiert seit 1981 und ist ein Event welches von der SAE in den USA gegründet und nach Europa „exportiert“ wurde. Mittlerweile findet dieses nahezu weltweit statt und wird nicht nur von den werdenden Ingenieuren geachtet. Längst wird es als Sprungbrett für Ingenieure in die Automobilindustrie und sogar den Rennsport gehandelt. Der Ansporn für die studentischen Teams liegt dabei darin nach wissenschaftlichen und wirtschaftlichen Gesichtspunkten einen Formelrennwagen ins Leben zu rufen. Mit diesen Fahrzeugen messen die Teams sich dann auf den Events in diversen statischen und dynamischen Aufgaben. Dabei ist für die teilnehmenden Teams, wie auch in der Industrie, eine stetige Weiter- als auch Neuentwicklung unumgänglich.

Der Autor dieser Bachelorarbeit ist ein Teil des Motorsportteams TMM der Hochschule Mittweida. Dieses besteht seit dem Jahr 2006. Nachdem in der Saison 2010/2011 mit dem Rennwagen „PHOENIX“ große Erfolge eingefahren wurden, schwor sich das Team in der Saison 2011/2012 an diese anzuknüpfen. Der nun in der aktuellen Saison entworfene Bolide namens „TIMM“ setzt dabei ein Ausrufezeichen. Er hat sich in diversen Tests schon jetzt als neuer Hoffnungsträger des Motorsportteams präsentiert. Der Rennwagen steht nicht nur durch die Integration der elektropneumatischen Schaltung sehr solide da. Auch der Renningenieur Thomas Sandmann rechnet „TIMM“ gute Chancen aus. Er konnte bei einem der ausgiebigen Tests anwesend sein, den neuen Boliden im Einsatz sehen und ihn durch verschiedene Aufgaben für die Fahrer und den Rennwagen auf Herz und Nieren prüfen. Auch nach Herrn Sandmanns Ansicht ist rein konstruktiv für die einzelnen Aufgaben der Events nicht viel mehr aus dem Rennwagen herauszuholen. Da auch das Team dieser Ansicht war und ist, wurde über die Integration weiterer elektronischer Systeme nachgedacht. Dabei entstanden neue Projekte wie die Entwürfe einer Automatikschaltung und einer Antriebsschlupfregelung. Diese beiden Systeme könnten den Rennwagen besonders während des Einsatzes bei einem Beschleunigungsrennen noch wesentlich weiter bringen. Da eine Entwicklung beider Systeme den Rahmen dieser Ba-

chelorarbeit sprengen würde, wurde sich dafür entschieden vorerst nur eine ASR zu entwickeln. Dieses kann im Gegensatz zur Automatikschaltung auch eigenständig agieren und bringt dem Team somit mehr Nutzen.

1.2 Zielsetzung

Die vorliegende Bachelorarbeit befasst sich mit dem Entwurf einer Antriebsschlupfregelung für einen Formula Student Rennwagen. Dies umfasst die Dimensionierung und Auswahl der Komponenten in Abstimmung mit der Installationsart, dem Installationsort und der Konstruktion der dafür geeigneten Geberscheiben. Zu beachten ist dabei der Frequenzbereich der einzelnen Bauelemente und die entstehende Frequenz durch die Geberscheibe und die Sensoren. Außerdem wird ein Programm für einen Microcontroller entworfen, welches diese Daten aufnehmen und verarbeiten soll. Am Ende der Bachelorarbeit steht ein Test des Systems.

2 Stand der Technik des TMM Rennwagens

Um einen besseren Einblick in die bestehende Problematik zu ermöglichen wird in diesem Kapitel der Stand der Technik beschrieben. Dabei wird ein besonderes Augenmerk auf die im Rennwagen verbauten Komponenten gelegt.

2.1 Antriebsstrang

Der Antriebsstrang besteht grundlegend aus den folgenden Elementen: dem Motorradmotor samt Getriebe, Ritzel und Kettenblatt, einer Motorradkette, den Antriebswellen und den Rädern. Im Folgenden werden die Bestandteile betrachtet, welche einen maßgeblichen Einfluss auf das Entstehen von Schlupf an den Hinterrädern haben (siehe Abbildung 1).

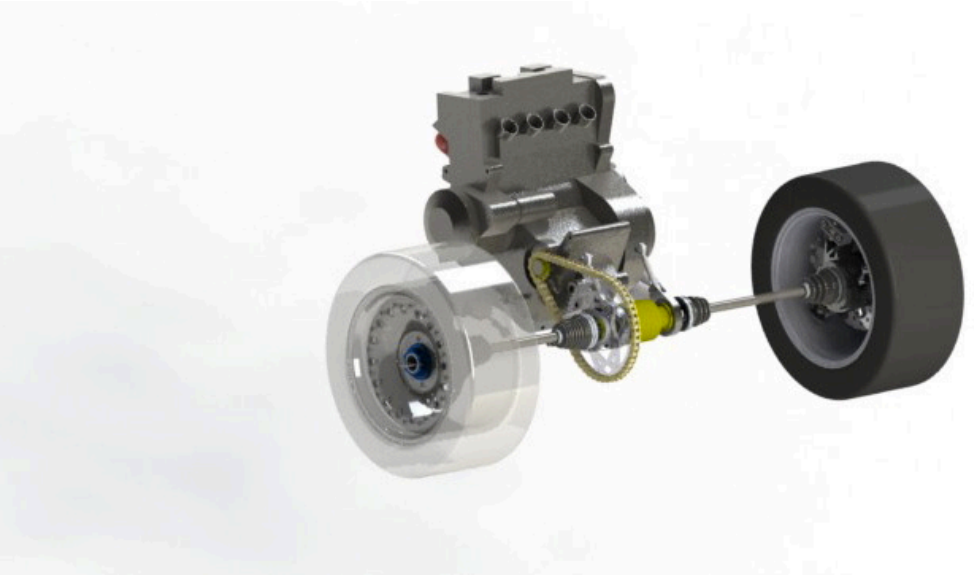


Abbildung 1: Antriebsstrang

2.1.1 Motor

Durch eine Einschränkung des Regelwerkes dürfen bei der Formula Student nur Motoren mit einem maximalen Hubraum von 610 ccm verwendet werden. Im Fall von TMM wurde daher ein Serienmotorradmotor einer Honda CBR 600 F (CBR 600 PC3, Bj. 1999) ausgewählt. Dieser bringt original bei einer Leistung von ca. 109 PS ein Drehmoment von ca. 62 Nm auf die Hinterachse (siehe Abbildung 2). Die Leistung dieses Motors wird dann noch einmal durch das Regelwerk begrenzt. Im gemeinsamen Ansaugtrakt, über welchen die Zylinder mit Luft versorgt werden, muss ein „restrictor“ eingesetzt werden. Dieser hat die Form eines Ringes, einen Durchmesser von 20 mm und schränkt die Luftzufuhr somit drastisch ein. Effektiv liegt der Rennwagen dann bei einer Leistung von ca. 70-80 PS. [FSAE12] [MOTO12] [TMM12]

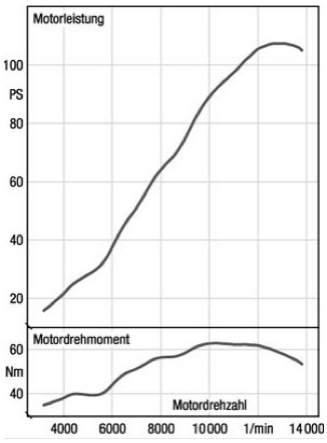


Abbildung 2: M-n-P-Kennlinie [MOTO12]

2.1.2 Zahnräder

Im derzeitigen Rennwagen wurde sich für ein Ritzel mit 11 Zähnen und einem Kettenblatt 50 Zähnen entschieden, wodurch sich ein Übersetzungsverhältnis (i) von 4,55 ergibt. Die Kraftübertragung erfolgt dabei über eine 525er Motorradkette. Um dem Schlupf etwas vorzubeugen wird derzeit noch am Entwurf des Kettenblattes gearbeitet.

$$i = \frac{Z_{Abtrieb}}{Z_{Antrieb}} = \frac{50}{11} = 4{,}55$$

2.1.3 Bereifung

Durch einen Sponsoringvertrag mit der Firma Continental erhält TMM pro Saison mehrere Sätze an Slicks und Regenreifen. Während Tests mit diesen Slicks und einem käuflich erworbenen Vergleichssatz der Firma Hoosier wurde festgestellt, dass der Bereich in dem die Continental Slicks Grip aufbauen unter normalen Rennbedingungen nur schwer zu erreichen ist. Im Gegenteil dazu funktionieren die Hoosier Slicks wesentlich zeitiger. Somit kann mit den Continental Slicks nicht das maximale/ideale Drehmoment auf die Fahrbahn gebracht werden um einen bestmöglichen Beschleunigungsvorgang zu gewährleisten.

2.2 Motorsteuergerät

Verwendet wird schon seit mehreren Saisons ein Motorsteuergerät der Firma Trijekt. Dieses ist frei programmierbar und eröffnet viele Möglichkeiten. Unter anderem enthält es beispielsweise einen Funktionseingang über welchen mehrere Parameter geändert werden können. Dieser ist in der Standartversion leider nicht nutzbar, er kann jedoch durch ein kostenpflichtiges Update freigeschaltet werden. Für die in dieser Arbeit behandelte Thematik ist auch ein separater Eingang am Motorsteuer vorhanden. Durch Ansteuerung dieses Eingangspins werden für einen vorbestimmten Zeitraum die Einspritzung und die Zündung unterbrochen. Genutzt wird dieser Eingang bereits beim hoch- als auch beim herunterschalten. Da aktuell noch hart geschalten wird, das heißt ohne zu kuppeln, wird durch die Zündunterbrechung die Beanspruchung auf das Getriebe und den gesamten Antriebsstrang gemindert.

3 Grundlagen und Vorbetrachtungen

In diesem Kapitel wird der grundlegende Bestandteil der behandelten Thematik erläutert. Des Weiteren werden Einflüsse aus dem Reglement der Formula Student aufgezeigt.

3.1 Definition ASR

Grundsätzlich sorgt das ASR dafür, dass beim Beschleunigen auf jeglichem Untergrund die antreibenden Räder nicht ungehindert durchdrehen. Dafür existieren in der Automobilindustrie verschiedene Lösungen. Der Grund dafür ist, dass es zwei dabei meist verwendete Systeme gibt. Zum einem wird ein Eingriff in das Motormanagement genutzt. Das heißt, sowohl die Zündung, als auch die Einspritzung werden für einen vordefinierten Zeitbereich ausgesetzt. Außerdem werden die Drosselwalzenstellung und der Zündwinkel verändert. Zum Anderen wird ein System genutzt, welches die Räder, die eine zu hohe Drehfrequenz aufweisen, durch gezielte kurze Bremsimpulse der Bremsanlage leicht abbremst. Die meisten ASR bestehen aus einer Kombination der beiden Systeme, teilweise werden sie jedoch auch einzeln verwendet. Die verwendeten Systeme der Automobilhersteller unterscheiden sich dabei meist nur in der Bezeichnung.

3.2 Definition Schlupf

Um näher zu beschreiben wann genau die ASR eingreift, wird im Folgenden der Schlupf beschrieben. Gebildet wird dieser aus der Differenz der Raddrehzahlen zwischen den Vorder- und Hinterrädern. Schlupf entsteht also genau dann, wenn die treibenden Räder eine höhere Drehfrequenz aufweisen als die angetriebenen Räder. Jedoch gilt zu beachten, dass der Schlupf bis zu einer gewissen Grenze sogar der Beschleunigung dienlich ist (siehe Abbildung 3). Das liegt an der übertragbaren Umfangskraft $F_{W,}$, welche mit zunehmendem Schlupf recht stark ansteigt. Das Maximum dieser Kurve liegt dabei bei relativ kleinen Schlupfwerten. Um die maximale Antriebskraft nutzen zu können, darf keines der antreibenden Räder dieses Maximum überschreiten. Mit steigender übertragbarer Umfangskraft nimmt jedoch die übertragbare Seitenkraft $F_{W,}$ ab. Für das ASR bedeutet das, dass am Maximum der Kurve der Regelbereich endet, um eine gewisse Spurtreue des Fahrzeuges aufrecht erhalten zu können. Die Abhängigkeit der Kennlinie ist dabei stark abhängig von den Reifen, der Fahrbahn und der Geschwindigkeit. [RWT08]

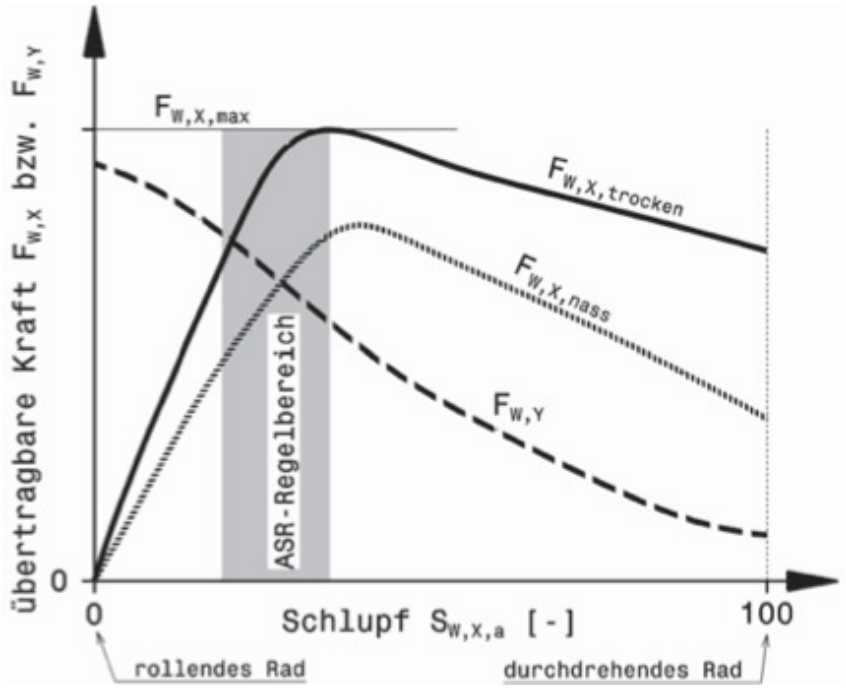


Abbildung 3: Regelbereich ASR [RWT08]

3.3 Funktion ASR

Bevor die ASR eingreifen kann, ist es nötig Daten aufzunehmen, speziell die Drehfrequenzen der Räder. Diese werden dabei von Sensoren aufgenommen, welche an den jeweiligen Rädern angebracht sind. Ermöglicht wird das durch Drehzahlgeberscheiben. Das Signal, die Drehfrequenz, gestaltet sich dabei nach Anzahl der Zähne der Drehzahlgeberscheibe. Diese Drehfrequenzen werden von einem Microcontroller aufgenommen und verglichen. Sind die treibenden Räder nun schneller als die angetriebenen Räder, so wird ein Eingriff in das Motormanagement eingeleitet. Dies sollte jedoch erst ab einem Schlupf von 8- 15% geschehen, da erst ab diesem Punkt die Umfangskraft ihr Maximum erreicht. Das Überschreiten dieser Grenze soll dabei durch das ASR verhindert werden. Alle Vorgänge müssen natürlich in Echtzeit abgefertigt werden, um das System sinnvoll einsetzen zu können. Vorrangig ist dies bei einem Beschleunigungsvorgang aus dem Stand der Fall. [RWT08] [WASR12]

3.4 Machbarkeitsanalyse und Hardware der ASR

Nach ersten Recherchen sind die Bestandteile, welche für das ASR nötig sind, ein Microcontroller, Sensoren zur Raddrehzahlerfassung und diverse Anschlussteile. Nach einer Analyse des Ausgangssignales der Sensoren könnten eventuell noch weitere Bauelemente nötig werden.

Die Sensoren, die allgemein zur Realisierung geeignet wären, werden nun kurz beschrieben. Unterschieden werden dabei zwei Gruppen, passive und aktive Sensoren, mit ihren jeweiligen Vertretern.

Passive Sensoren:

Induktive Sensoren, welche nach dem Prinzip der Induktion arbeiten, reagieren auf elektrisch leitende Körper. Der Sensor generiert dabei durch einen internen Oszillator (Schwingkreis: Spule und Kondensator) ein elektromagnetisches Feld. Die dabei auftretende Oszillatorspannung wird durch das Eindringen eines elektrisch leitfähigen Körpers, in das elektromagnetische Feld, verändert/geschwächt. Wird dabei ein gewisser Grenzwert überschritten, wird der Ausgang aktiv geschalten. [LM12]

Kapazitive Sensoren arbeiten auf der Grundlage der Änderung der Kapazität eines Kondensators. Der Sensor selbst hat dabei zwei aktive Elemente, eine Sensorelektrode und eine Abschirmung. Nähert sich diesem ein Körper, welcher aus elektrisch leitenden als auch nichtleitenden Material bestehen kann, so erfolgt eine Änderung der Kapazität. Durch eine Kapazitätszunahme beginnt der im Sensor befindlicher Oszillator (Schwingkreis: Widerstand und Kondensator) zu schwingen. Dieses Schwingen wird erkannt, der Ausgang wird dadurch aktiv geschalten. [LM12]

Aktive Sensoren:

Hall-Sensoren gehören zu der Gruppe der Magnetfeldsensoren. Diese arbeiten mit Feldern von Elektro- oder Dauermagneten. Der Sensor wertet dabei die sogenannte Hallspannung U_H aus, welche durch die Lorentzkraft entsteht. Die Lorentzkraft beschreibt dabei die Ablenkung eines Elektronenstromes durch ein Magnetfeld. Dadurch verarmt eine Seite an Elektronen, auf der anderen Seite werden Elektronen angereichert. Die dabei entstehende Spannung ist die Hallspannung. Ab einem voreingestellten Grenzwert wird der Schaltausgang aktiv geschalten. [LM12]

$$U_H = R_H \cdot \frac{B \cdot I}{s}$$

$R_H - Hallkoeffizient$
$B - magnetische\ Flussdichte$
$I - Stromstärke$
$s - Leiterdicke$

Lichtschranken bestehen grundsätzlich aus einem Sender und einem Empfänger, welche ein elektronisch-optisches System bilden. Der Sender ist dabei eine Lichtstrahlenquelle, der Empfänger ein Sensor, meist ein Phototransistor. Das Schaltsignal wird bei Lichtschranken durch eine Unterbrechung des Lichtstrahles realisiert. Der Sensor erkennt diese Unterbrechung und schaltet den Ausgang aktiv. [LSW12]

3.5 Definition Echtzeit

„Gemäß DIN 44300 wird unter Echtzeit beziehungsweise Realbetrieb der Betrieb eines Rechnersystems verstanden, bei dem Programme zur Verarbeitung anfallender Daten ständig betriebsbereit sind, derart, dass die Verarbeitungsergebnisse innerhalb einer vorgegebenen Zeitspanne verfügbar sind. Die anfallenden Daten oder Ereignisse können je nach Anwendungsfall nach einer zufälligen zeitlichen Verteilung oder zu bestimmten Zeit-

punkten auftreten“ (EZS05, S. 1). Echtzeitsysteme, auch zeitkritische Systeme genannt, setzen sich zusammen aus diversen Hard- und Softwarekomponenten. Allgemein erfassen diese Systeme interne und externe Daten sowie Ereignisse und verarbeiten diese zugleich. Essentiell ist dabei eine zeitrichtige Übermittlung der Ereignisse der Informationsverarbeitung an weitere Systeme oder Prozesse. Die Abarbeitung folgt einer vorgegebenen Strategie, das heißt asynchron beziehungsweise zyklisch. Als externe Ereignisse beschreibt man Beispielsweise den Takt eines externen Zeitgebers oder eine Sensoranforderung. Interne Ereignisse sind Hard- und Softwareinterrupts. Ein Interrupt ist dabei eine kurzfristige Unterbrechung des normalen Programmablaufes. Somit steht hinter dem Begriff Echtzeit zum einem ein korrektes Ergebnis, als auch eine Erfüllung der Zeitbedingungen. [EZS05]

3.6 Definition Abtasttheorem

Um ein Signal in digitaler Form, mit allen Informationen, aus einem analogen Signal zu erhalten, muss man das Abtasttheorem nach Nyquist beziehungsweise Shannon berücksichtigen. Folgendes sagt es aus: „Sind in einem Signal $x(t)$ die Frequenzen in einem Band von $0 - f_{max}$ vorhanden, so reicht es, das Signal $x(t)$ in zeitlichen Abständen $T_0 = 1/(2\,f_{max})$ abzutasten, um aus der Funktion x_N die ursprüngliche Größe $x(t)$ ohne Verlust an Informationen zurückgewinnen zu können.“ (EZS05, S. 91f). Das heißt, das Signal muss mit $f_{abtast} > 2\,f_{Signal}$ abgetastet werden. Wird das Signal mit einer zu niedrigen Frequenz abgetastet, so entsteht der Aliasing-Effekt. Das bedeutet, es werden durch falsche Abtatsung Fehler in das digitalisierte Signal eingebracht. [EZS05] Dies ist in der Abbildung 4 zu erkennen. Sie zeigt, dass ab einer Grenze von $f_{abtast} = 2\,f_{Signal}$ mit sinkender Abtastfrequenz das detektierte Signal immer mehr vom „original“ analogen Signal abweicht. Angewendet auf das vorliegende Projekt muss der Microcontroller das Eingangssignal der Sensoren wenigstens mit der beschrieben Abtastfrequenz, wenn nicht sogar höher, abtasten beziehungsweise abfragen.

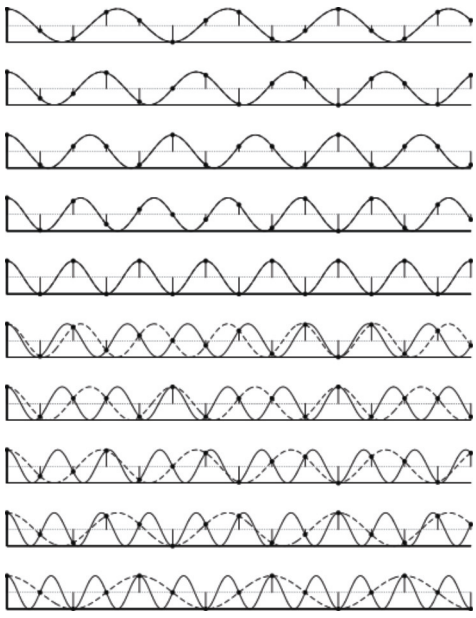


Abbildung 4: Erhöhung der Signalfrequenz über die halbe Abtastfrequenz [WNSA12]

4 Projektplan

In Kapitel 4 soll der schrittweise Ablauf des Projektes dargestellt und aufgelistet werden. Um eine optimale Bearbeitung des Projektes gewährleisten zu können, wurde im Vorfeld ein Projektablaufplan erstellt (siehe Abbildung 5). Aus diesem wird ersichtlich, dass 2 Projektuntergruppen entstehen. Zum einem die „Auswahl, Dimensionierung & Beschaffung der Komponenten“, „Analyse über Installationsort & -art der Sensoren“ und „Entwurf & Fertigung der Signalgeber“. Als auch der „Entwurf des Microcontrollerprogrammes“ und die „Installation & Test des Systems“. In den folgenden Punkten werden nun die Bestandteile der einzelnen Projektuntergruppen behandelt.

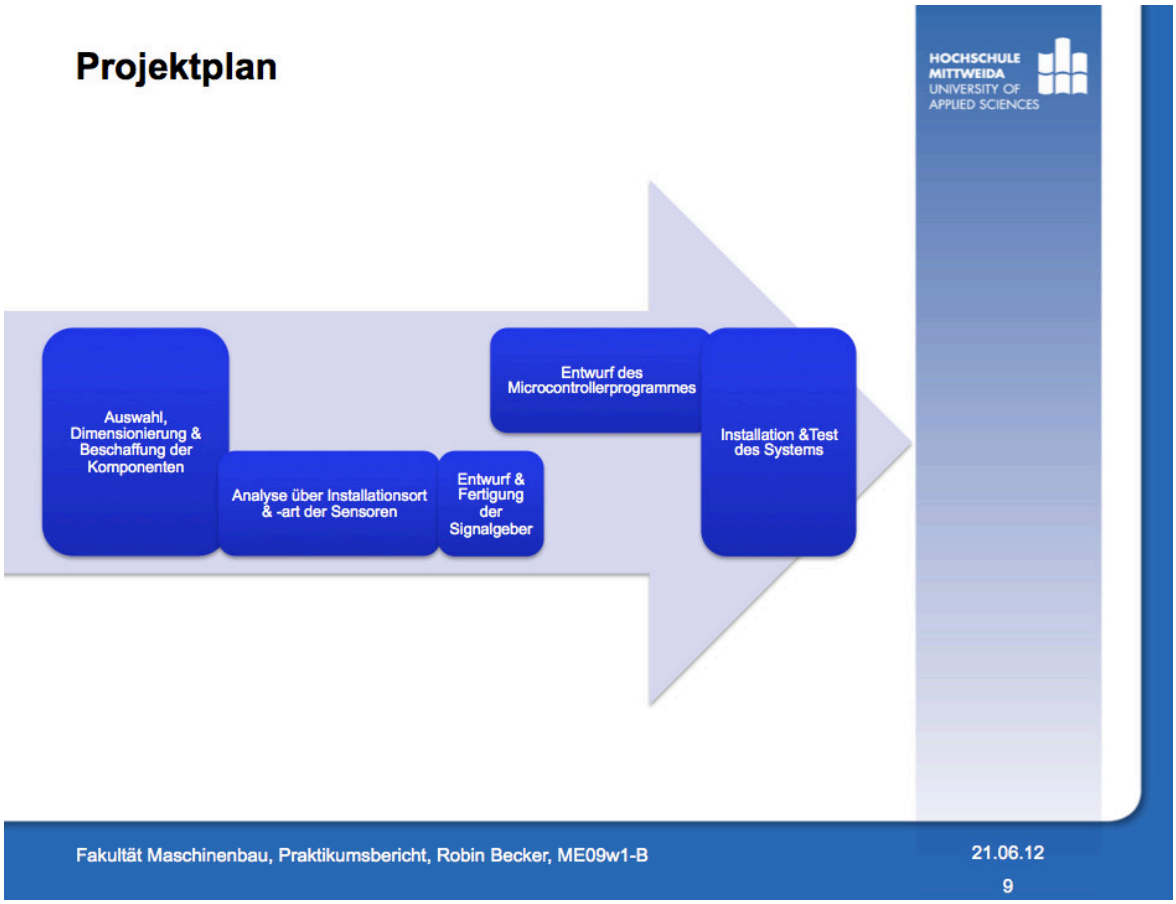


Abbildung 5: Projektplan

4.1 Auswahl, Dimensionierung & Beschaffung der Komponenten

Bei den zu beschaffenden Komponenten handelt es sich um Sensoren zur Erfassung der Raddrehzahl und einem Microcontroller, welcher diese Informationen verarbeiten soll. Dabei muss die passende Art von Sensor für die vorliegende Aufgabe herausgesucht werden. Dabei stehen kapazitive, induktive, optische und Hall-Sensoren zur Auswahl.

Es muss darauf geachtet werden, dass die Sensoren witterungsbeständig, als auch vibrationsbeständig sind und im nötigen Frequenzbereich arbeiten. Der Frequenzbereich wird maßgeblich durch die Signalgeberscheibe vorgegeben. Zur Dimensionierung sind Installationsart und Installationsort mit zu beachten. Auch muss berücksichtigt werden, dass der Microcontroller die Masse an Informationen in Echtzeit verarbeiten kann.

4.2 Installationsort & Installationsart der Sensoren

Um eine einwandfreie Funktion der Sensoren zu erreichen, muss die für sie optimale Position herausgefunden werden. Auch muss beachtet werden, dass eine problemlose Verlegung der Kabel möglich ist. Um Fehlsignalen vorzubeugen, ist der Kontakt mit Fluiden jeglicher Art zu vermeiden.

4.3 Entwurf und Fertigung der Signalgeber

Auch im Fall der Signalgeber ist auf den Installationsort und die Installationsart zu achten. Aus identischem Grund wie bei den Sensoren, ist ein Kontakt mit Fluiden jeglicher Art zu vermeiden. Die Anzahl der „Zähne“ der Signalgeber ist zu bestimmen. Dies erfolgt anhand der benötigten beziehungsweise möglichen Frequenz.

4.4 Entwurf des Microcontrollerprogrammes

Zur Verarbeitung der Signale der Raddrehzahlsensoren muss ein Programm für den ausgewählten Microcontroller geschrieben werden. Dieses soll die jeweiligen Drehzahlen der einzelnen Räder miteinander vergleichen. In dem Fall, dass die Antriebsräder in einem bestimmten Maße schneller drehen als die Vorderräder, soll eine Zündunterbrechung erfolgen um dem Durchdrehen der Hinterräder Einhalt zu gebieten.

4.5 Test des Systems

Bevor das System als Ganzes getestet wird, ist anzustreben, dass vorerst die einzelnen Komponenten getestet werden. Unterteilt werden die Tests daher in vier Gruppen. Dem Test der Sensoren samt Ansteuerung für die Microcontrollereingänge, dem Programmtest rein im Simulator, dem Test des Microcontrollers mit Eingangsbeschaltung per Funktionsgeneratoren und zuletzt der Test des kompletten Systems. Erst nach erfolgreichem Abschließen dieser Tests kann das ASR im Rennwagen verbaut werden.

5 Systemkonzept

Kapitel 5 befasst sich mit allen vorbereitenden Aufgaben des Projektes, das heißt mit der Auslegung, Auswahl und dem Entwurf der verwendeten Komponenten.

5.1 Auswahl der Bauelemente

In diesem Punkt des Systemkonzeptes wird die Reihenfolge die Vorgehensweise bei der Auswahl und der Positionierung der Bauelemente beschrieben.

5.1.1 Auswahl des Installationsortes und der Installationsart

Da das ASR später am bestehenden Rennwagen zum Einsatz kommen soll, galt es besonders den Installationsort mit diesem abzustimmen. Von vornherein wurde sich dafür entschieden so wenig wie möglich neue Bauelemente hinzuzufügen. Vielmehr galt es bestehende Bauelemente des Rennwagens umzufunktionieren.

An den Vorderrädern war schnell eine Lösung gefunden. Im Radträger der Vorderräder werden die beiden Radlager von einem Distanzring, welcher sich in deren Mitte befindet, auseinander gehalten. Dieser kann auch als Geberscheibe ausgelegt werden. Vorteil dessen ist, dass die Radlager gekapselt sind. Das heißt, Schmutz und Nässe können die Messung nicht beeinflussen. Um den Sensor anbringen zu können, muss eine Bohrung in den Radträger, direkt über dem Distanzring, eingebracht werden. Der Sensor selbst muss dann mit Schraubensicherung und einer Mutter befestigt werden. Abbildung 6 zeigt schematisch den angestrebten Aufbau.

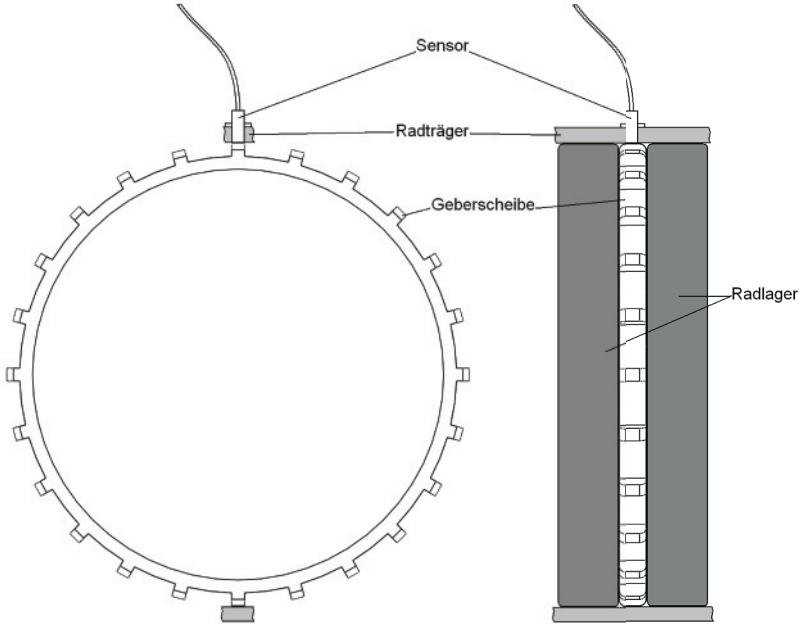


Abbildung 6: Schema Drehzahlgeberscheibe Vorderräder

An den Hinterrädern musste eine andere Lösung gefunden werden, da dort kein Distanzring zwischen den Radlagern im Radträger sitzt. Jedoch werden die Radlager durch einen Distanzring von außen, über eine Mutter, im Radträger arretiert, welcher für die gegebene Aufgabe gut geeignet ist. Auch dieser Distanzring muss konstruktiv umgestaltet werden, die Hauptgeometrie bleibt jedoch bestehen. Der Sensor wird dann über dem Distanzring im Radträger, ähnlich der Variante der Vorderräder, positioniert. Bei dieser Variante ist das System nicht komplett gekapselt. Jedoch ist das Eintreten von Schmutz und Wasser recht unwahrscheinlich, da zwischen Mutter und Radträger nur ein Spalt von ca. 1 mm ist. Der Sensor selbst wird in diesem Fall, geometriebedingt, nur mit Schraubensicherung befestigt.

5.1.2 Auswahl der Sensoren

Begonnen wurde bei der Auswahl mit einer Internetrecherche. Untersucht wurden dabei folgende Punkte: die allgemeine Art, der Frequenzbereich (Schaltfrequenz), die Größendimension, die Bauform, die Betriebsspannung, die Ausgangsschaltung, der Nennschaltabstand und die Schutzart des Sensors.

Bei der Art des Sensors boten sich vier Varianten. Unterschieden werden musste dabei zwischen den passiven Sensoren, wie induktiven Sensoren und kapazitiven Sensoren und aktiven Sensoren wie Hall-Sensoren und Lichtschranken (optische Sensoren). In der Automobilindustrie werden für ähnliche Aufgaben meist induktive Sensoren oder Hall-Sensoren gewählt.

Nach der Reihenfolge der Kriterien der Internetrecherche wurde dann der passende Sensor ausgesucht. Hauptgrund dafür, warum in der Automobilindustrie meist auf induktive Sensoren oder Hall-Sensoren zurück gegriffen wird, liegt im möglichen Frequenzbereich, also der Schaltfrequenz der Sensoren. Die mögliche Schaltfrequenz bei kapazitiven Sensoren liegt meist bei ca. 100 Hz und ist somit nicht ausreichend für die geforderte Aufgabenstellung. Die Schaltfrequenz bei induktiven Sensoren, optischen Sensoren, als auch bei Hall-Sensoren, liegt zwischen 1 KHz bis zu 20 KHz. Durch die Größendimension und die möglichen Bauformen fiel schlussendlich die Wahl auf einen induktiven Sensor. Dieser ist im Gegensatz zum Hall-Sensor als Miniaturausführung erhältlich, das heißt als M4 Ausführung. Aufgrund des vorliegenden Installationsraumes ist dies ideal. Des Weiteren kann der ausgewählte Sensor bündig eingebaut werden und besitzt die Schutzart IP 67. Daher kann er auch in einem Rennfahrzeug, welches diversen Witterungsbedingungen ausgesetzt wird, installiert werden. Das ist auch der ausschlaggebende Punkt, weshalb eine Miniaturlichtschranke nicht in die nähere Auswahl kommt. Diese wäre viel zu anfällig für äußere Bedingungen, wie Feuchtigkeit und Schmutz. Das Manko der Hall-Sensoren ist, dass das zu detektierende Objekt magnetisch sein muss und der Entwurf der Geberscheibe somit einen enorm erhöhten Aufwand und auch erhöhte Kosten hätte. Diese müsste in diesem Fall komplett magnetisiert oder mit magnetischen Zähnen ausgestattet werden. Eine detaillierte Übersicht dazu ist in Tabelle 1 zu sehen.

Tabelle 1: Vergleich der Sensoren

Sensorart	induktiver Sensor	kapazitiver Sensor	Hall- Sensor	Lichtschranke
Frequenzbereich	bis 3KHz	bis 100Hz	bis 20KHz	bis 5KHz
Größendimension	Miniatur, M4	Miniatur, M5	Standard, M8	Miniatur, 15,5 x 13,4 x 6,4
Bauform	zylindrisch mit Gewinde	zylindrisch mit Gewinde	zylindrisch mit Gewinde	Gabellichtschranke, rechteckig
zu detektierendes Material	elektrisch leitfähige Objekte	Metalle und Nichtmetalle	Elektro- und Dauermagneten	Metalle und Nichtmetalle
Betriebsspannung	10...30 VDC	10...30 VDC	10...30 VDC	5...24 VDC
Einbauart	bündig	bündig	bündig in nichtmagn. Metall	nicht bündig
Ausgangsschal-tung	PNP/NPN/ NAMUR	PNP/NPN/ NAMUR	PNP/NAMUR	PNP
Nennschaltab-stand	0...1mm	0...1mm	10...60mm	Gabelweite = 5mm
Schutzart	IP67	IP67	IP67	IP40
Arbeitstemperatur	-25...+75°C	-25...+75°C	-25...+75°C	-25...+55°C

Nachdem eine Auswahl an Herstellern herausgesucht wurde, welche den gewünschten Sensortyp produzieren, wurden von diesen Angebote eingeholt. Schlussendlich wurde eine Anzahl von vier Stück der induktiven Sensoren mit der Bezeichnung IFRM 04P15B1/L bei dem Sensorenhersteller Baumer bestellt. Das Angebot und das Datenblatt des ausgewählten Sensors sind im Anhang beigefügt (siehe Anlage 1 und Anlage 2).

5.1.3 Auswahl des Microcontrollers

Bei der Auswahl des Microcontrollers boten sich auch mehrere Alternativen. Der erste Blick fiel auf ein evaluation Board der Firma Arduino, den Arduino Nano. Dieses wird als praxisnah beschrieben und soll einen guten Einstieg ermöglichen. Im Laufe eines Expertengespräches mit Prof. Dr.-Ing. Hagenbruch der Fakultät EIT der Hochschule Mittweida wurde ein Board mit einem Atmel Xmega Microcontroller empfohlen. Hintergrund für die Empfehlung war, dass die Atmel Xmega Microcontrollerreihe über Timer/Counter mit einer „capture“ Funktion von 4 Eingangskanälen verfügt. Das heißt Werte, die die Eingänge liefern, werden in gewissen Zeitzyklen aufgenommen und können dann verarbeitet werden. Andere Microcontroller verfügen ebenfalls über Timer/Counter mit einer Capture Funktion, diese erlauben dann jedoch meist nur eine Aufzeichnung von maximal zwei Kanälen. Im weiteren Verlauf wurde daraufhin das evaluation Board Atmel XMEGA-A1 Xplained als Alternative ausgewählt. Ein evaluation Board ist dabei eine mit einem Microcontroller bestückte „Prototypenplatine“, welche über periphere Elektronik, insbesondere Schnittstellen, verfügt. Darauf zu achten war, dass der ausgewählte Microcontroller in der Lage sein muss die Masse an Informationen der Sensoren in Echtzeit aufzu-

nehmen und zu verarbeiten. Nach dem Vergleich der beiden Varianten wurde sich für das Atmel XMEGA- A1 Xplained entschieden. Selbst wenn die Capture-Funktion nicht den gewünschten Erfolg bringen sollte, so ist man mit dem ausgewählten Board trotzdem in der Lage ein alternatives Programm zu realisieren. Großer Vorteil des Atmel-Boards ist dabei die 32 MHz clock speed. In Tabelle 2 sieht man die Unterschiede der beiden evaluation Boards.

Tabelle 2: Vergleich der evaluation Boards

Board:	Arduino UNO	Atmel XMEGA- A1 Xplained
Microcontroller:	ATmega328	ATxmega128A1
clock speed:	16MHz	32MHz
digital I/O Pins:	14	24
analog Pins:	6	8
operating Voltage:	5V	5V oder 2,7 - 3,6V
Weiteres:	Tutorials vorhanden Programmierung via USB Port	8 Taster und 8 LED´s onboard
Kosten (ohne Versand):	29,95 €	36,26 €

5.2 Auslegung der Geberscheiben

Bei der Auslegung der Geberscheiben galt es auf die gegebenen Randbedingungen zu achten. Diese wurden gebildet durch den ausgewählten Sensor mit einer maximalen Schaltfrequenz von 3 KHz, dem Microcontroller mit 32 MHz und der maximalen Geschwindigkeit von ca. 150 Km/h. Daher wurde dazu eine Berechnung aufgestellt. Diese wurde anschaulich in einer ausführlichen Berechnung dargestellt, welche der Arbeit im Anhang beigelegt ist (siehe Anlage 3). Geometrisch wurden die Drehzahlgeberscheiben so ausgelegt, dass der Sensor über ihnen steht.

5.2.1 Auslegung nach der maximalen Rennwagengeschwindigkeit

Um die maximale Drehzahl zu erhalten, welche der Sensor erkennen soll, wurde diese aus der maximalen Geschwindigkeit des Rennwagens und dem Umfang der Räder errechnet.

$$n_{max} = \frac{v_{max}}{u} = \frac{v_{max}}{\pi \cdot d}$$

$n_{max} - maximale\ Raddrehzahl$
$v_{max} - maximale\ Fahrzeuggeschwindigkeit$

Um die maximale Raddrehfrequenz zu erhalten, die der Raddrehzahlsensor auslesen kann, benötigt man eine vorgegebene Anzahl der Zähne der Geberscheibe.

$$f_{max} = \frac{(z \cdot n)}{60} = \frac{(z \cdot n_{max})}{60}$$

Dabei erhält man für z = 1: $f_{max} = n_{max}$

$f_{max} - maximale\ Drehfrequenz$
$z \quad - Zähnezahl\ der\ Drehzahlgeberscheibe$

Die Überschlagsrechnung wurde für Polling und somit einem zyklischen Programmablauf ausgelegt. Um zu erkennen, ob der Microcontroller in der Lage ist die Menge an Daten aufzunehmen und zu verarbeiten wurden überschlagsmäßig die Rechenschritte pro Schaltflanke im Pollingbetrieb ermittelt.

$$Rechenschritte\ pro\ Schaltflanke\ (RpS) = \frac{Clsp}{(i \cdot z \cdot 2 \cdot f_{max} \cdot k)}$$

$Clsp - clock\ speed\ des\ Microcontrollers$
$i \quad - \ Anzahl\ der\ Raddrehzahlsensoren$
$k \quad - Korrekturfaktor$

Der Korrekturfaktor wurde der Berechnung hinzugefügt, um das ungleiche Größenverhältnis zwischen den Zähnen und Lücken der Geberscheiben auszugleichen. Hintergrund dafür ist das Abtasttheorem, das auf ein ausgeglichenes Größenverhältnis zwischen den Zähnen und Lücken ausgelegt ist.

Im folgenden Ausschnitt werden die Ergebnisse der Berechnung aufgezeigt. In Tabelle 3 werden die Zähnezahl der Drehzahlgeberscheibe, die maximale Drehfrequenz und die Rechenschritte pro Schaltflanke des Microcontrollers dargestellt. Die Rechenschritte pro Schaltflanke geben dabei an wie viele Rechenschritte zur Verarbeitung einer Schaltflanke beziehungsweise eines Zahnes verfügbar sind.

Tabelle 3: Berechnung RpS bei v_{max}

z (Zähnezahl)	f_{max} [Hz]	RpS
1	27	24691
2	54	12346
4	108	6173
8	216	3086
12	324	2058
16	432	1543
20	540	1235
24	648	1029
32	864	772
48	1296	514

Die Tabelle zeigt, dass selbst bei der Verwendung einer Geberscheibe mit 48 Zähnen, die 3 KHz-Frequenzgrenze des ausgewählten Sensors nicht überschritten wird. Auch erkennt man, dass der Microcontroller ausreichend Ressourcen für die Aufnahme und die Verarbeitung der Daten hat. In diesem Fall würde die Wahl trotzdem auf eine Geberscheibe mit 24 Zähnen fallen, um noch genügend Sicherheiten zu haben.

5.2.2 Auslegung nach der ASR

Allgemein muss die Frage gestellt werden, bis zu welcher Geschwindigkeit der Eingriff einer ASR sinnvoll beziehungsweise nötig ist. Bei PKWs agiert das ASR meist im unteren Geschwindigkeitsbereich. Erfasst man über die Geberscheibe die Raddrehzahl und ermittelt daraus die Geschwindigkeit, so kann man dies auch im Rennwagen realisieren. Man benötigt aber diesen Vergleichswert und eventuell auch noch den momentan verwendeten Gang. Die Grenze für den Eingriff der ASR wird dabei auf 40 Km/h festgelegt. Setzt man diese Geschwindigkeit nun in die Berechnung, wie auch unter 5.2.1, ein, so erhält man folgende Tabelle (Tabelle 4). Die ausführliche Berechnung ist zu finden unter Anhang 4.

Tabelle 4: Berechnung RpS bei v_{ASR}

z (Zähnezahl)	f_{ASR} [Hz]	RpS
1	8	83333
2	16	41667
4	32	20833
8	64	10417
12	96	6944
16	128	5208
20	160	4167
24	192	3472
32	256	2604
48	384	1736

$f_{ASR} - \ maximale\ Drehfrequenz\ f\ddot{u}r\ das\ Eingreifen\ \ des\ ASR$

Auch in Tabelle 4 sieht man, dass sowohl die Sensoren, als auch der Microcontoller kein Problem mit den auftretenden Frequenzen hätten.

5.2.3 Varianten der Geberscheiben

Um bei einem späteren Test auf eventuelle Probleme reagieren zu können, wurde sich dafür entschieden vorerst nur eine Varianten der Geberscheiben zu entwerfen und zu fertigen. Gewählt wurde dabei die Variante mit 24 Zähnen, da diese noch genügend Sicherheiten bietet. Auch wird die ASR vorerst auf die maximale Rennwagengeschwindigkeit ausgelegt.

5.3 Vordefinierung der Elektronik

Als Vorgabe der Formula Student gilt, dass alle elektrischen/elektronischen Bauelemente im Rennwagen durch die Betätigung von einem der Not-Aus-Schalter am Fahrzeug spannungsfrei zu schalten sind. Daher ist für das ASR kein weiterer Not-Aus-Schalter von nöten. Jedoch wird vorgesehen das ASR durch einen extra Schalter ein- und ausschalten zu können. Auch soll eine Anzeige per LED realisiert werden, welche das Eingreifen der ASR anzeigt. Zum optischen Test der Sensoren soll vom Microcontroller pro Sensor eine LED als Funktionsanzeige geschaltet werden. Des Weiteren muss eine Möglichkeit der dynamischen Spannungswandlung ausgewählt und entworfen werden, welche das Schaltsignal der Sensoren von 12 V auf die 3,3 V Eingangsspannung des Microcontrollers wandelt. Um einer Beeinflussung durch elektrische oder elektromagnetische Effekte durch andere elektrische Geräte vorzubeugen, müssen die Leitungen der Sensoren geschirmt werden.

5.4 Beschaffung der Bauelemente

Nachdem die Vorbetrachtung und Auslegung abgeschlossen waren, wurde damit begonnen die benötigten Bauelemente zu beschaffen. Dazu wurden für die beiden kostspieligeren Posten mehrere Angebote eingeholt. Der Anbieter mit dem besten Preis-/Leistungs-Verhältnis wurde ausgewählt. Zudem musste eine Bedarfsanforderung für die ausgewählten Bauelemente mit dem ausgewählten Anbieter an die Hochschule gestellt werden. Eine Auflistung der entstehenden Kosten ist in Tabelle 5 zu sehen.

Tabelle 5: Gesamtkosten

Anbieter	Preis pro Stück in €	Rabatt in %	Kosten für 4 Sensoren in €	Gesamtkosten (ohne Versand etc.) in €	Förmliches Angebot liegt vor
Baumer	68,7	35	178,62	178,62	ja
Ballufff	63	10	226,8	226,8	nein
Pepperl & Fuchs	0	0	0	0	nein
Schlüter	69	5	262,2	262,2	ja
Reichelt Elektronik	41,55	0		41,55	ja
Watterott	39,2	0		39,2	ja
Gesamtpreis =	251,7578 €		(Verpackung, Versand und MwSt. inbegriffen)		

Die angenommenen Angebote liegen dem Anhang bei (Anlage 1 und Anlage 4).

6 Umsetzung

In diesem Kapitel wird die Vorgehensweise bei der Umsetzung des Projektes beschrieben.

6.1 Entwurf der Geberscheiben

Nachdem die Auslegung der Geberscheiben abgeschlossen wurde, konnte mit deren Konstruktion begonnen werden. Die bestehenden Bauelemente, welche in den vergangenen Saisons bereits über Solid Works entworfen wurden, dienten dabei als Vorlage. Die Geometrie wurde für die benötigten Bedingungen angepasst. Die Zahnhöhe und die Zahnbreite wurden anhand der Sensordaten ausgelegt. Die Zahnhöhe, als auch die Zahnbreite betragen 2,00 mm. Die Zahnhöhle ergab sich dabei aus dem Nennschaltabstand des Sensors, dieser beträgt 0,8 mm (siehe Anlage 2). Um ein einwandfreies, eindeutiges Signal zu erhalten, wurde für den Nennschaltabstand 1 mm angenommen und dieser verdoppelt. Die geometrische Beschaffenheit der Zähne der Drehzahlgeberscheiben kann je nach Fertigungsart von den im Folgenden beschrieben Drehzahlgeberscheiben abweichen. Die Grundmaße bleiben dabei bestehen, lediglich die Steilheit der Flanken könnte anders ausfallen. Abbildung 7 zeigt links den Idealfall und rechts eine eventuelle fertigungsbedingte Abweichung. Die Fertigungszeichnungen der entworfenen Drehzahlgeberscheiben liegen dem Anhang bei (Anlage 5 und Anlage 6). Das verwendete Material für die Geberscheiben wird in beiden Fällen von den Distanzringen übernommen. Diese wurden aus Aluminium , EN AW 7075, gefertigt.

Abbildung 7: Abweichung Flankensteilheit

6.1.1 Geberscheiben für die Vorderräder

Der bestehende Distanzring gab bei dem Entwurf die Maße der neuen vorderen Geberscheiben vor. Dessen Außendurchmesser betrug 56,00 mm, der Innendurchmesser 50,20 mm und die Breite 4,50 mm. Des Weiteren wurden die Kanten des Distanzringes um 1 mm verrundet, um dessen Einsetzen zu vereinfachen. Das einzige Maß, welches dabei nicht übernommen wurde, war der Außendurchmesser. Dieser muss zwar in gewissen Grenzen bleiben, jedoch nicht den exakten Wert des alten Distanzringes haben. Um die gewünschte Zahnhöhe zu erreichen wurde der Außendurchmesser auf 58,2 erhöht. Die somit entstehende Geberscheibe wird in Abbildung 8 dargestellt. Abbildung 9 und Abbildung 10 zeigen die Radträger mit dem Sensor und der Geberscheibe (gelb gekennzeichnet) im 3D Modell.

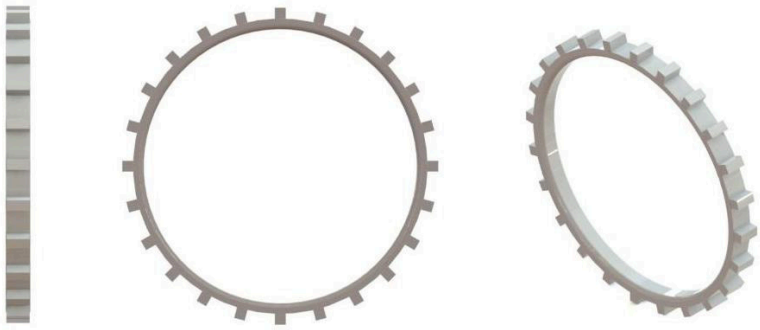


Abbildung 8: Drehzahlgeberscheiben vorn

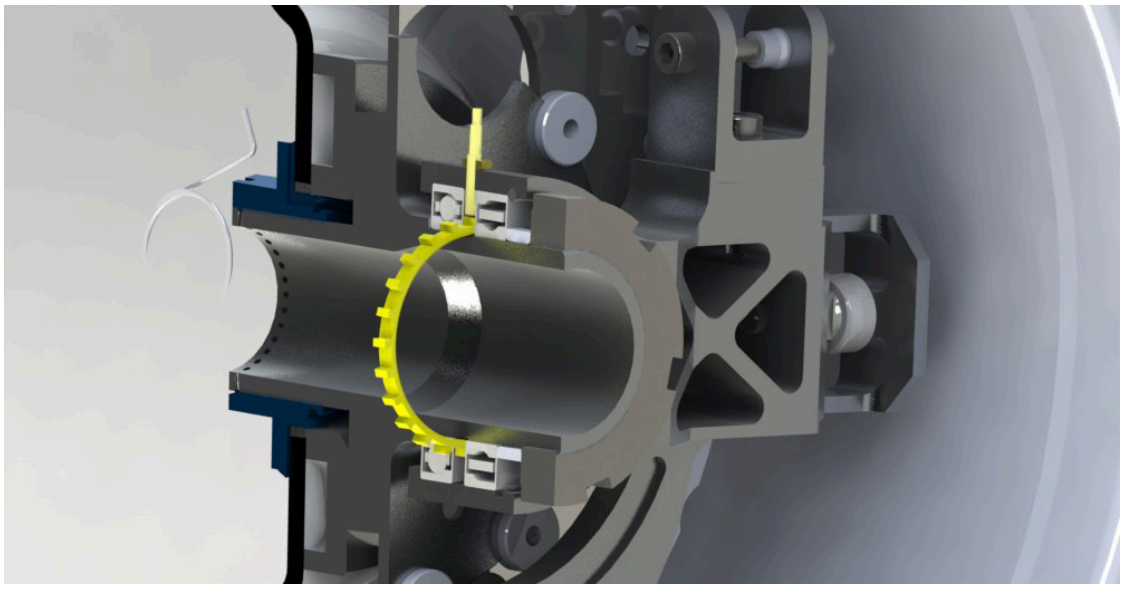


Abbildung 9: Radträger vorn links im Schnitt (90 Grad nach links gedreht)

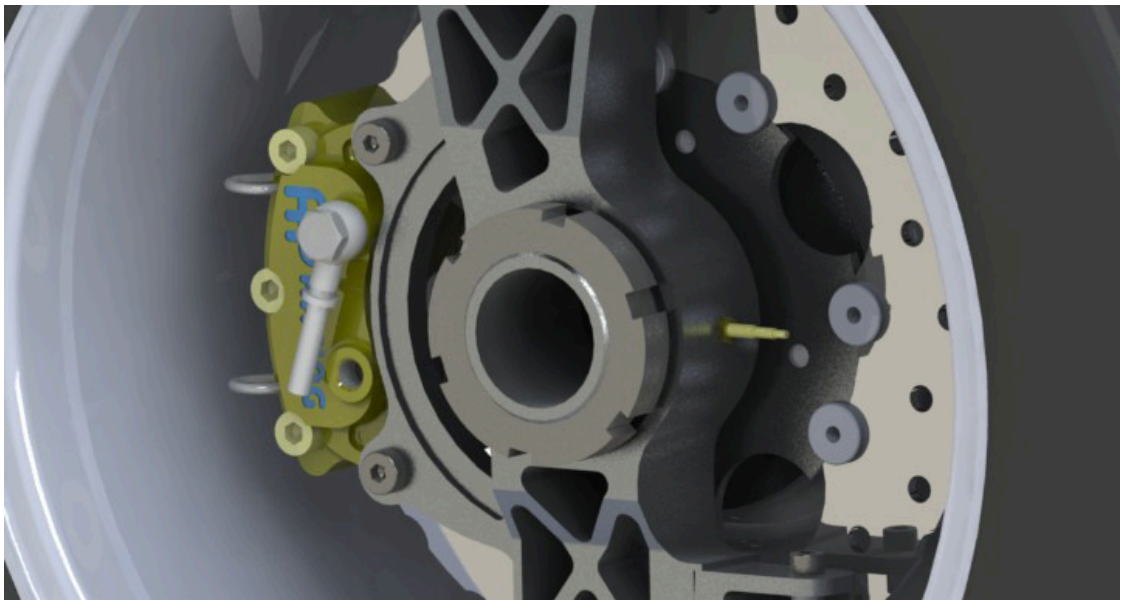


Abbildung 10: Radträger vorn links Komplett mit Sensor

6.1.2 Geberscheiben für die Hinterräder

Auch in diesem Fall wurden die Maße des bestehenden Distanzringes übernommen. Dessen Außendurchmesser betrug 80,00 mm, der Innendurchmesser 75,00 mm und die Breite des Ringes 12,00 mm. Durch die Verzahnung, die der Geometrie hinzugefügt wurde, erhöhte sich der Außendurchmesser auf 85,00 mm. Der Innendurchmesser und die Breite wurden beibehalten. Die Kanten wurden mit einer Fase von 2x2 mm abgeschrägt, um naheliegende Bauelemente nicht zu beeinflussen. Die daraus entstehende Geberscheibe ist in Abbildung 11 zu sehen. In Abbildung 12 und Abbildung 13 ist der Radträger hinten links mit Sensor und Drehzahlgeberscheibe (gelb gekennzeichnet) im 3D Modell zu sehen.

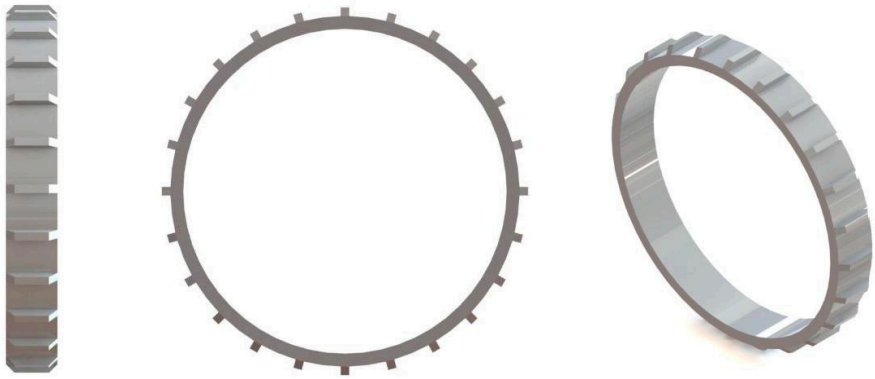


Abbildung 11: Drehzahlgeberscheiben hinten

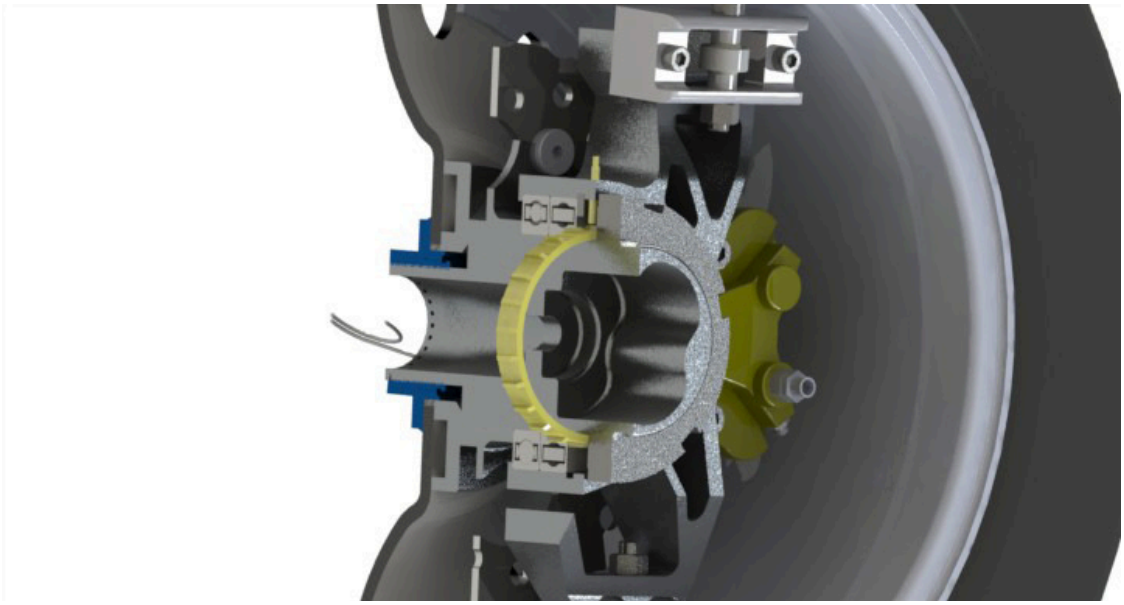


Abbildung 12: Radträger hinten links im Schnitt

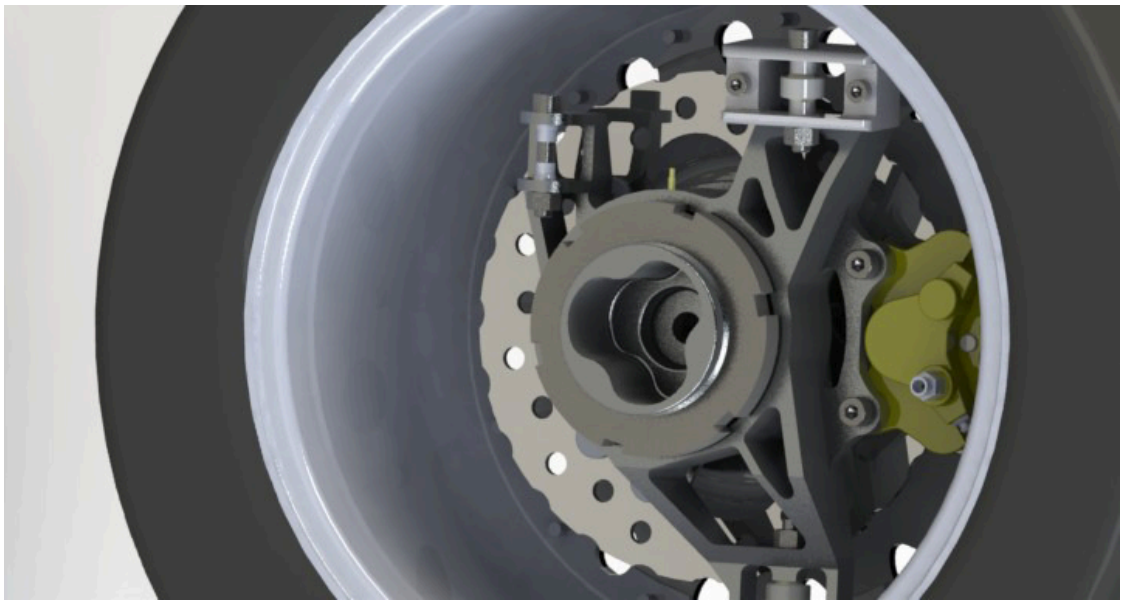


Abbildung 13: Radträger hinten links komplett mit Sensor

6.2 Vorbereiten der Elektronik

Da die ausgewählten Sensoren bei Aktivierung ein Schaltsignal ausgeben, das deren Versorgungsspannung (12V) entspricht und für das Atmel Xmega Xplained Board für die Digitaleingänge maximal V_{cc}, die Versorgungsspannung (3,3V), zulässig ist, musste dafür noch eine Lösung gefunden werden. Anderenfalls würde eine Missachtung dessen zur Beschädigung oder Zerstörung des evaluation Boards führen. Ausgewählt wurde dafür pro Kanal ein Spannungsteiler, welcher das Schaltsignal des jeweiligen Sensors auf ca. 3,3 V begrenzt. Abbildung 14 zeigt den Schaltplan der entworfenen Schaltung. Dem Spannungsteiler wurden eine Zener-Diode (Z-Diode) und ein Kondensator beigefügt. Die Zener-Diode, mit einer Z-Spannung (U_Z) von 3,3 V, soll dabei Spannungsspitzen im statischen Bereich am Eingang des Microcontrollers verhindern. Die Z-Spannung wird auch als Durchbruchspannung bezeichnet und beschreibt genau den Punkt an dem die Z-Diode in Sperrrichtung ab U_Z niederohmig wird, also durchbricht. Dies stabilisiert die Spannung im vorliegenden Spannungsteiler auf die gewünschte Ausgangsspannung von 3,3 V. In Durchlassrichtung ist ihr Verhalten gleich dem normaler Dioden. Der Kondensator mit einer Kapazität von 100 pF soll Spannungsspitzen im dynamischen Bereich entgegenwirken. Er bildet mit dem Widerstand R_1 einen RC-Tiefpass, welcher höher frequentiere Anteile im Eingangssignal abschwächt. Die Kapazität wurde so gering wie möglich gehalten, um den Spannungsteiler nicht zu beeinflussen. Um die Eignung der Schaltung für die vorliegende Problemstellung feststellen zu können, wurde im Folgenden die Grenzfrequenz berechnet.

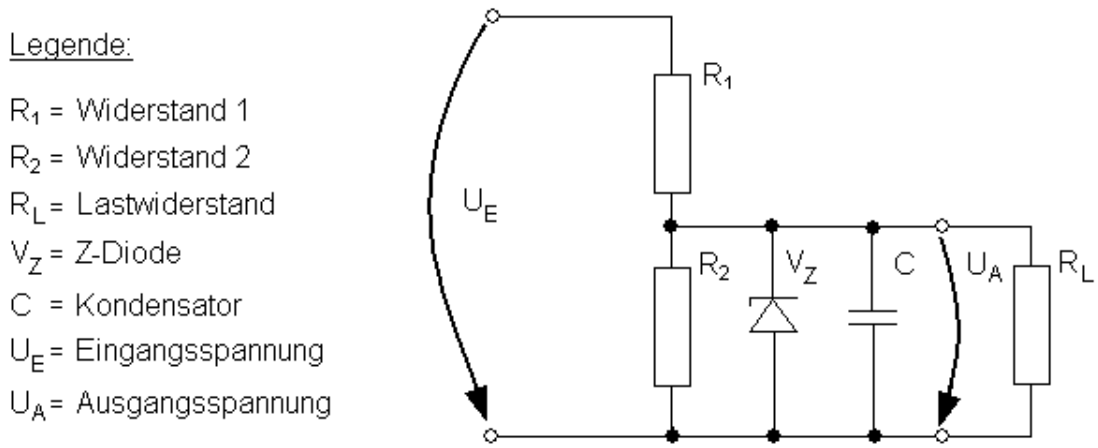


Abbildung 14: Schaltplan Spannungsteiler

Der Spannungsteiler allgemein wurde wie folgt berechnet:

Parallelwiderstand:

$$R_P = \frac{R_2 \cdot R_L}{R_2 + R_L} = \left(\frac{620 \cdot 1000}{620 + 1000}\right)\Omega = 383\Omega$$

Ausgangsspannung:

$$U_A = \frac{R_P}{R_1 + R_P} \cdot U_E = \frac{383\Omega}{1000\Omega + 383\Omega} \cdot 12V = 3{,}3V$$

Gesamtwiderstand:

$$R_{ges} = R_1 + R_P = 1000\Omega + 383\Omega = 1383\Omega$$

Grenzfrequenz:

$$f_g = \frac{1}{2\pi RC} = \frac{1}{2\pi \cdot 1383\Omega \cdot 100\text{pF}} = 1151031Hz \approx 1{,}15MHz$$

$$\left[1pF = 1 \cdot 10^{-12}F = 1 \cdot 10^{-12}\frac{S}{\Omega}\right]$$

Die Berechnung legt dar, dass der Spannungsteiler bei den auftretenden Frequenzen problemlos mithalten kann. Abbildung 15 und Abbildung 16 zeigen, dass trotz des zusätzlichen Kondensators keinerlei Auswirkung auf das Ausgangssignal des Spannungsteilers sicht- und messbar ist. Die Aufnahmen entstammen einer Oszilloskopmessung mit einem ähnlichen Aufbau. Die Ausgangsspannung betrug dabei 5,1 V, welches durch eine Z-Diode mit einer Z-Spannung von 5,1 V begrenzt wurde. Es lässt sich gut erkennen, dass bei steigender Eingangsspannung, die Ausgangsspannung maximal den Wert der Z-Spannung, also der Durchbruchspannung der Z-Diode, annimmt. Besonders gut zu erkennen bei der Messung mit $U_E = 30\,V$, da dort die Spannung bei 5,1 V abgeschnitten wird, das heißt die Z-Diode durchbricht.

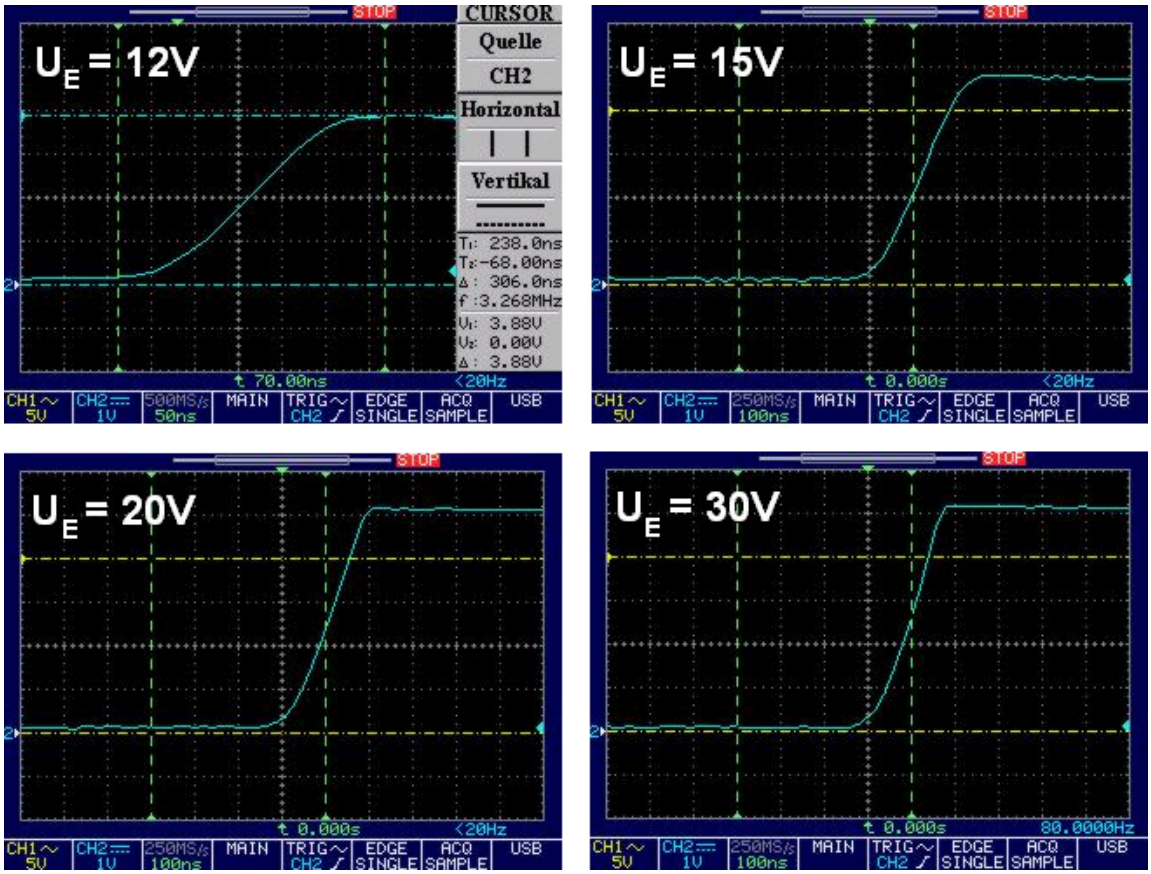


Abbildung 15: Oszilloskopaufnahme Spannungsteiler ohne C

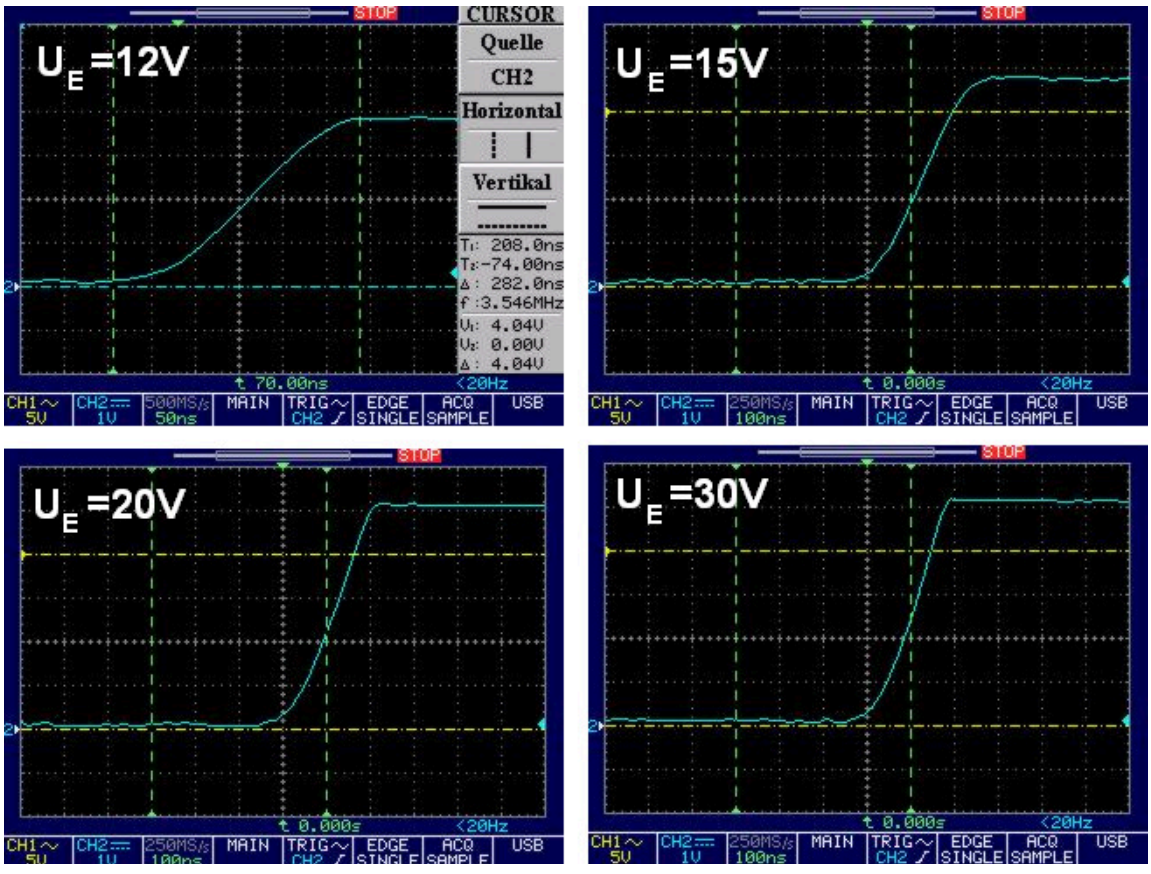


Abbildung 16: Oszilloskopaufnahme Spannungsteiler mit C

Der Versuchsaufbau, welcher für die Aufnahmen genutzt wurde, ist in Abbildung 17 zu sehen. Rechts erkennt man, dass dies der Aufbau für die Aufnahme mit dem Spannungsteiler ohne Kondensator ist.

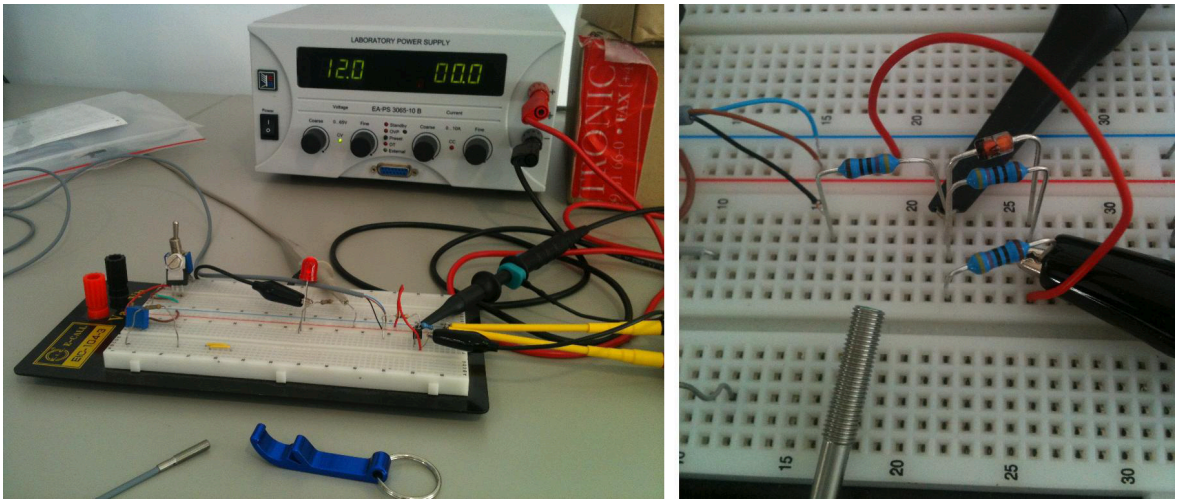


Abbildung 17: Versuchsaufbau Spannungsteiler

6.3 Programmierung des Microcontrollers

Der Quellcode des Microcontrollers sollte in der Hochsprache C geschrieben werden. Der Hintergrund dessen war zum einem, die weite Verbreitung dieser, zum anderem der fortgeschrittene Wissensstand des Autors in dieser Programmiersprache. Als Programmieroberfläche wurde das AVR Studio 4 ausgewählt. Bevor jedoch mit der Programmierung begonnen werden konnte, galt es sich in den Microcontroller ATXmega128A1 und in das Evaluation Board Xplained einzulesen und einzuarbeiten.

6.3.1 Vorbereitende Arbeitsschritte

Um effektiv mit dem Microcontroller arbeiten zu können, wurden vor der Programmierung diverse Unterlagen, wie das Manual der ATxmega A Reihe, Application notes und Datasheets von der Internetseite des Hersteller Atmel besorgt. Die Herausforderung dabei lag darin die vielen Informationen der einzelnen Dokumente zusammen zu tragen und diese bestmöglich zu nutzen. Unterschiede zwischen der Dokumentation und der vorliegenden Hardware des Xplained128A1- Boards sollten sich im weiteren Verlauf als Problem herausstellen. Nach studieren des Schaltplan des Boards konnte dieses jedoch ausgemerzt werden.

Für einen einfachen Test des Boards wurden zu Beginn die Ein- und Ausgänge getestet, welche Hardwareseitig direkt auf dem Board mit Tastern und LEDs bestückt waren. Nach Konfiguration der dazu nötigen PORTs und PINs und einer Programmzeile, welche das Bitmuster der Eingänge den Ausgängen zuweist, konnte das erste Programm getestet werden (siehe Abbildung 18).

```
//***************************************************************
// LED und Taster

#define LED PORTE                               //LED -> PORTE
#define BUTTOND PORTD                           //BUTTOND -> PORTD
#define BUTTONR PORTR                           //BUTTONR -> PORTR
#define MASK_1 0b11000000                       // Maske 1 fuer Buttons PORTD 0-5
#define MASK_2 0b11111100                       // Maske 2 fuer Buttons PORTR 6-7, vor schieben
#define MASK_3 0b00111111                       // Maske 3 fuer Buttons PORTR 6-7, nach schieben
#define PINCONF 0b00011001                      // Pin Config fuer alle Buttons 0(SRLEN) 0(INVEN) 011(OPC) 001(ISC)

int main(void)
{

        LED.DIRSET = 0b11111111;                //LED als Ausgänge setzen
        BUTTOND.DIRSET = 0b00000000;            //Buttons PORTD als Eingänge setzen
        BUTTONR.DIRSET = 0b00000000;            //Buttons PORTR als Eingänge setzen
        BUTTOND.PIN0CTRL = PINCONF;
        BUTTOND.PIN1CTRL = PINCONF;
        BUTTOND.PIN2CTRL = PINCONF;
        BUTTOND.PIN3CTRL = PINCONF;
        BUTTOND.PIN4CTRL = PINCONF;
        BUTTOND.PIN5CTRL = PINCONF;
        BUTTOND.PIN6CTRL = PINCONF;
        BUTTONR.PIN0CTRL = PINCONF;
        BUTTONR.PIN1CTRL = PINCONF;


        while(1)                // Endless loop
        {

                LED.OUT = (BUTTOND.IN | MASK_1) & (((BUTTONR.IN | MASK_2) << 6) | MASK_3);
                // Inputmuster zusammenfassen & auf Ausgang setzen
        }       // end while
}       // end main

//***************************************************************
```

Abbildung 18: Testprogramm mit Tastern und LED´s

Um das Board ohne Debugger bespielen zu können, muss im Vorfeld noch ein USB Treiber für das Board, als auch eine Toolchain zum überspielen des Quellcodes installiert werden. Will man das Board bespielen, so muss beim Anstecken des USB-Kabels der SW0 Taster gedrückt gehalten werden bis die Status-LED aufhört zu blinken (Abbildung 19).

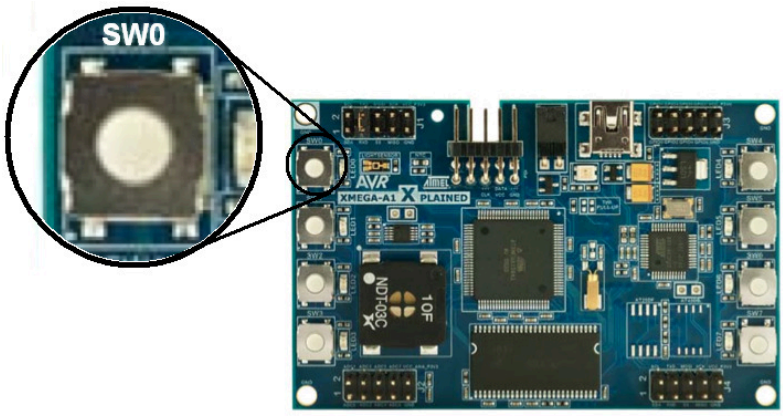


Abbildung 19: Xplained Board SW0 Taster

Bevor mit dem Schreiben des ASR Quellcodes begonnen werden konnte, wurden die dafür verwendeten Ein- und Ausgänge festgelegt (siehe Abbildung 20), als auch ein Ablaufdiagramm (Abbildung 21) für die vorliegende Problematik entworfen.

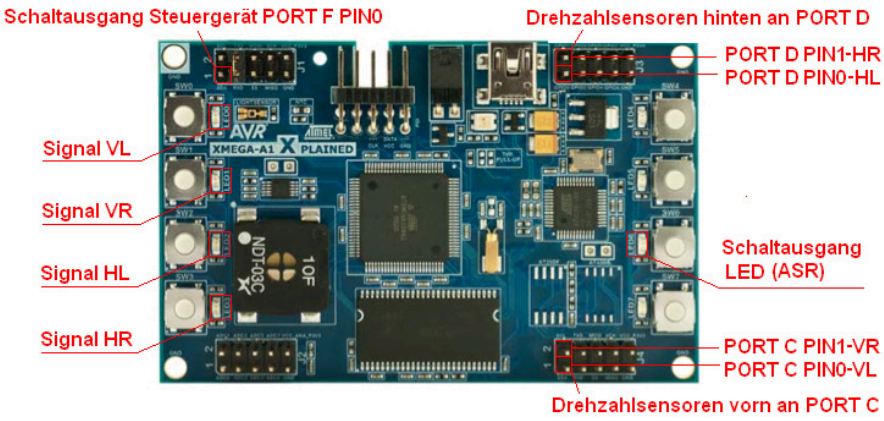


Abbildung 20: Xplained Board, Ein- und Ausgangsdefinition

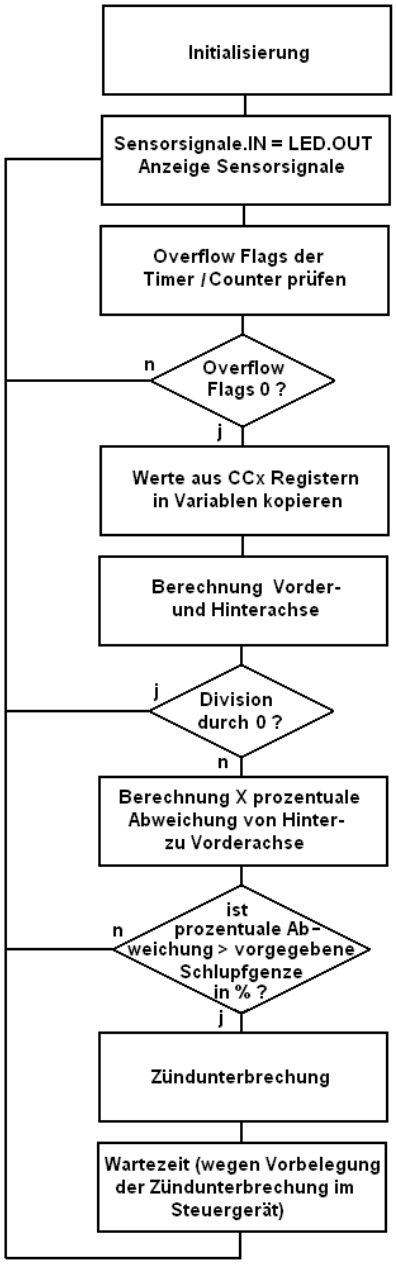


Abbildung 21: Ablaufdiagramm ASR

6.3.2 Schreiben des ASR Quellcodes

Um die Ressourcen des Microcontrollers voll ausschöpfen zu können, wurde sein Systemtakt von 2 MHz auf 32 MHZ umgestellt. Dies wurde über ein Unterprogramm realisiert, welches der "XMEGA Timer/Counter extension driver example source" von Atmel entnommen wurde. Im Kopf des Programmes unter „Variablen Deklaration“ werden alle im Programm verwendeten Variablen initialisiert und teilweise vorbelegt. Diese wurden als *unsigned integer* 32 Bit festgelegt, um eventuellen Überträgen bei den Berechnungen von vornherein ausschließen zu können. Die Variable X wurde als einzige als *integer* definiert, da bei dem Vergleich, nach der Berechnung von X, ein negativer Wert durchaus sinnvoll ist. Die aus dem Testprogramm gewonnenen Informationen dienten direkt als Einstieg in das eigentliche Programm für die ASR. Die Konfiguration für die Eingangspins der Sensoren, als auch der Ausgänge wurde übernommen und musste nur noch angepasst werden (vgl. Anhang 7). Im weiteren Verlauf der Arbeit werden „n“ und „x“ für eine Zahl zwischen 0...7 verwendet. Diese entspricht der Adresse eines Bits in einem 8 Bit Muster. Um die PINs eines PORTs als Ausgang zu schalten, muss das Register DIR (Data Direction Register) per DIRSET für den jeweiligen PIN an der jeweiligen Stelle im Bitmuster mit 1 belegt werden. Um alle PINs des Registers als Ausgänge zu schalten, muss DIRSET mit 0b11111111 belegt werden (Bsp. PORTn.DIRSET = 0b11111111;). Will man diese jedoch als Eingänge nutzen, so muss man DIRSET mit 0b00000000 belegen. Um die PINs nur auf steigende Flanken reagieren zu lassen, müssen diese im PINn Configuration Register (PINnCTRL) eingerichtet werden. Die Bits 0 bis 2 dienen dabei der Input/Sense Configuration. Für steigende Flanke (rising edge) ist dabei 001 einzutragen. Das Register wird dann wie folgt belegt: PORTn.PINnCTRL = 0b00000001. Auch pull-up oder pull-down Charakteristiken lassen sich in diesem Register einstellen. Für das vorliegende Programm wurden die Eingänge als pull-down geschalten. Die Flankeneinstellung lässt das Programm nur auf steigende Flanken des Eingangssignales reagieren. Fortgefahren wurde dann mit der Konfiguration der Timer/Counter, welche für den Modus *frequency capture* benötigt wurden. Der *frequency capture* ist dabei nur ein wählbarer Modus neben 7 weiteren Optionen, welche die Timer/Counter der Xmega A Baureihe bieten. Dieser Modus ermöglicht es direkt die Periodendauer eines Eingangssignales zu messen (siehe Abbildung 22). Die Zeit zwischen den steigenden Flanken ist dann die Periodendauer des Signales. [ATX09] [ATD09]

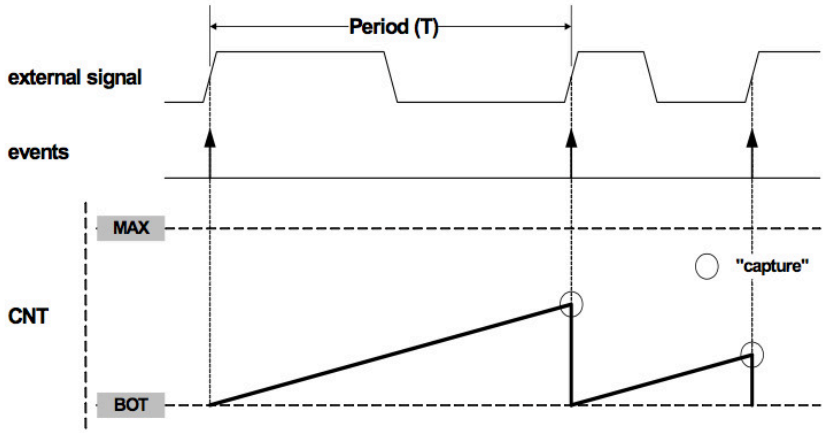


Abbildung 22: frequency capture [ATX09]

Als Initialisierung dient dabei das Sensorsignal, dessen Einstellung über den PIN bereits vorgenommen wurde. Ein großer Vorteil der capture Funktion ist, dass Flags aus dem INTFLAGS Register des Timer/Counter bei jeder steigenden Flanke neu gesetzt beziehungsweise zurückgesetzt werden. Wurde beispielsweise bei einer Aufnahme ein overflow erkannt und das overflow Flag gesetzt, so wird dieses bei erfolgreicher Folgeaufnahme wieder automatisch zurückgesetzt. Auch der Zählwert und die aufgenommene Zeit werden bei jeder Folgeflanke zurückgesetzt und neu gestartet. Da die Timer/Counter in dieser Einstellung nur per Event aktiviert werden können, muss dem jeweiligen Eingang noch ein Eventkanal zugewiesen werden. Das wird mit dem Register EVSYS.CHnMUX realisiert. Der jeweilige Timer/Counter muss dann noch für „capture or compare“ freigeschaltet werden. Außerdem muss eine Zähler TOP-Wert Einstellung vorgenommen und eine Taktquelle, mit oder ohne Prescaler, ausgewählt werden. Ein Prescaler dient der Skalierung der aufgenommenen Signale. [ATX09] [ATD09]

Berechnung des Prescalers für ein 16 Bit Register:

Der Prescaler muss so eingestellt werden, dass bei minimaler Frequenz des Signals, also bei größtmöglich erwarteter Zeit, zwischen den Signalen des Sensors, der Zählwert im Rahmen des maximalen Zählwertes liegt. Außerdem müssen die Werte bei hohen Frequenzen trotzdem noch gut unterscheidbar sein. Tabelle 6 zeigt, dass die Genauigkeit in höheren Frequenzbereich mit größerem Prescaler extrem sinkt. Deswegen wurde der Prescaler DIV256 ausgewählt. Folgende Formel wurde zur Berechnung verwendet:

$$Takte = {}^{CPU\_Takte}/_{Eventintervall}\Big/_{Prescaler}$$

$$Maximaler\ Zählwert\ (TOP - Wert)\ des\ Timer/Counter = 2^{16} - 1 = 65535$$

Tabelle 6: Berechnung Prescaler

Eventintervall		SYSCLK (CPU_Takt)	PRESCALE =clk/n	Takte
f_{min} =	2 Hz	32 MHz	256	62500
f_{max} =	650 Hz	32 MHz	256	192
f_{theor} =	1000 Hz	32 MHz	256	125
f_{min} =	2 Hz	32 MHz	1024	15625
f_{max} =	650 Hz	32 MHz	1024	48
f_{theor} =	1000 Hz	32 MHz	1024	31

Eingestellt wird der Prescaler mit folgender Eingabe: TCCn.CTRLA = 0b00000110, die Bits 110 stehen dabei für den ausgewählten Prescaler – clk/256.

An sich bieten die Timer/Counter die Möglichkeit die Signale von 4 Kanälen (A,B,C,D) aufzunehmen. Negativ daran ist, dass eine Routine zum Wechsel der Kanäle geschrieben werden müsste. Der Grund dafür ist, dass jedem Timer/Counter (TCxn) nur ein Zähler hinterlegt ist. Der Timer würde bei jeder steigenden Flanke den Zähler neu starten.

So kann man zwar alle Kanäle gleichzeitig aufnehmen, jedoch könnten diese Aufnahmen nicht genutzt werden, da dass eigentliche Signal unbrauchbar wird. Daher wurde pro Signaleingang jeweils ein Timer/Counter verwendet. Im Programm wieder zu finden als TCC0, TCC1, TCD0 und TCD1. Die genauen Inhalte und Register der Konfiguration, der Timer/Couner für jeden der 4 Kanäle, sind dem kommentierten Quellcode im Anhang zu entnehmen (siehe Anlage 7). [ATX09] [ATD09]

Nach erfolgreicher Umsetzung aller Initialisierungen und Voreinstellungen folgt nun der eigentliche Programmablauf. Der Hauptteil des Programmes läuft in einer Dauerschleife. Im Kopf dieser folgt eine weitere Schleife, welche zur Überprüfung der Eingangssignale, als auch als Sicherheitsschleife dient. Bedingung für den Ausstieg aus der Schleife ist, dass die overflow Flags aller verwendeten Timer/Counter 0 sind. Wenn eines der overflow Flags, der Timer/Counter gesetzt ist, so verweilt das Programm in dieser Schleife. Ist keines der Flags gesetzt, so wird die Schleife nur normal durchlaufen beziehungsweise abgearbeitet. Dies dient als Schutz bei dem Blockieren einzelner Räder oder dem Ausfall eines Sensors. Ohne diese Schleife würde ein Timer/Counter im Störfall, wie oben beschrieben, ein falsches Signal ausgeben, welches dann auch in die Berechnung eingehen würde. Da die internen LEDs der Sensoren durch die Installation in den Radträgern teilweise nur schwer zu sehen sind, werden deren Signale über LEDs auf dem Xplained Board, bei Durchlaufen der Sicherheitsschleife, wiedergegeben. Die folgende Legende zeigt die im weiteren Text verwendeten Variablen.

$VL - Periodendauer\ vorn\ links$	$HL - Periodendauer\ hinten\ links$
$VR - Periodendaier\ rechts\ vorn$	$HR - Periodendauer\ hinten\ rechts$
$V\ -$ Periodendauer Vordererachse	$H\ -$ Periodendauer Hinterachse
$S\ - Schlupfgrenzwert\ in\ Prozent$	$X\ - Schlupf\ in\ Prozent$

Nach Verlassen der Sicherheitsschleife werden die Werte, beziehungsweise Aufnahmen, der CCA Register der Timer/Counter in die jeweiligen Variablen für VL, VR, HL und HR gespeichert. Nachdem die Werte in die Variablen übertragen wurden, werden diese nun für die Berechnung für Vorder- und Hinterachse verwendet. Dabei wird jeweils der Mittelwert der Aufnahmen einer Achse gebildet. Hintergrund dessen ist, dass bei Kurvenfahrten das kurveninnere Rad schneller dreht als das Kurvenäußere. Durch die Mittelwertbildung pro Achse soll dies wieder ausgeglichen werden. Die Mittelwertbildung wurde durch eine Addition der beiden aufgenommenen Werte und eine folgende Division durch 2 realisiert. Bsp.:

$$V = {(VL + VR)}/{2}$$

Sind die Mittelwerte gebildet, erfolgt die Berechnung der prozentualen Abweichung zwischen Vorder- und Hinterachse, dem Schlupf X. Da eine eventuelle Division durch „0“ an dieser Stelle zu Problemen führen würde, wird die Berechnung nur ausgeführt wenn der

Divisor H nicht „0“ ist. In der dafür genutzten „if- Anweisung“ wird die Berechnung wird wie Folgt durchgeführt:

$$X = \frac{(V \cdot 100\%)}{H} - 100$$

Mit einer weiteren „if- Anweisung“ wird überprüft ob X größer als der Schlupfgrenzwert S, welcher mit 15 Prozent vordefiniert wurde, ist. Wenn die Aussage wahr ist, so wird der Schaltausgang für die Zündunterbrechung des Steuergerätes, als auch eine LED zur Visualisierung, geschalten. Dies erfolgt 30 ms lang, welches durch eine „for-Schleife“ (1. Wartezeit) mit einem Durchlauf von 30 Zyklen und delay wert von 1 ms realisiert wurde. Danach werden Schaltausgang und LED wieder zurückgesetzt. Da das Steuergerät eine definierte Zeit für die Dauer der Zündunterbrechung vorgibt und ein zu häufiges Schalten des Schaltausganges zum abwürgen des Motors führen würde, folgt eine weitere Wartezeit (2. Wartezeit). Auch diese wurde per „for-Schleife und einem delay von 1 ms realisiert, jedoch mit einem Durchlauf von 250 Zyklen. Das heißt, die Zündung kann durch das ASR maximal vier mal pro Sekunde unterbrochen werden. Nach Ablauf der 2. Wartezeit und verlassen der 2 „if-Anweisungen“ gelangt das Programm zum Fuße der Dauerschleife und beginnt wieder von vorn.

6.4 Test der ASR

Bis zur Abgabe der vorliegenden Arbeit wurde das System noch nicht komplett getestet, die einzelnen Systeme jedoch schon. So wurde ein Test der Sensoren mit dem entworfenen Spannungsteiler erfolgreich durchgeführt, siehe 6.2. Auch das Atmel Xplained evaluation Board wurde mit dem entworfenen Programm getestet. Dazu wurde der Versuchsaufbau in Abbildung 23 vorgenommen.

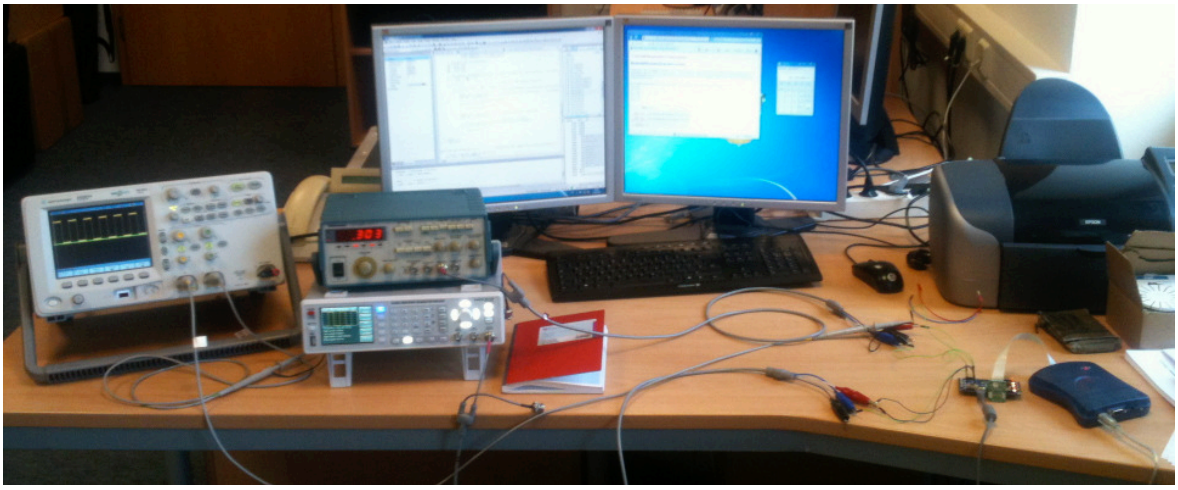


Abbildung 23: Versuchsaufbau Xplained Boards mit Funktionsgenerator

Für jede Achse (V, H) wurde ein Funktionsgenerator an die entsprechenden Pins angeschlossen. Zuvor wurde die Ausgangsspannung beider Funktionsgeneratoren per Oszilloskop auf die für die Eingänge nötige Spannung von 3,3 V und die Ausgabe einer Rechteckspannung eingestellt. Zum debuggen wurde ein Atmel JTAGICE mkII Debugger ge-

nutzt (siehe Abbildung 24). Mit diesem konnte der genaue Programmstand, als auch die Inhalte der Register überprüft werden. Um die Inhalte der Variablen per „Mouseover“ sehen zu können, wurde in der „Variablen Deklaration“ als Vorsatz „volantile“ verwendet. [ATX09] [ATD09]

Mit dieser Vorarbeit konnte das Programm getestet werden. Dabei galt es alle eventuellen Möglichkeiten durchzuspielen. Neben dem Normalbetreib H = V, wurde der Fall H > V (das Auftreten von Schlupf) und H < V (das Auftreten von negativen Schlupf) bei verschiedenen Frequenzen getestet. Dadurch das X durch seine Definition im Programmkopf einen negativen Wert annehmen kann und dieser dann folglich nicht größer als S sein kann, wird die Zündung bei H < V nicht unterbrochen. Dadurch konnten noch kleinere Fehler ausgemerzt und der Funktionstest schlussendlich erfolgreich durchgeführt und abgeschlossen werden.

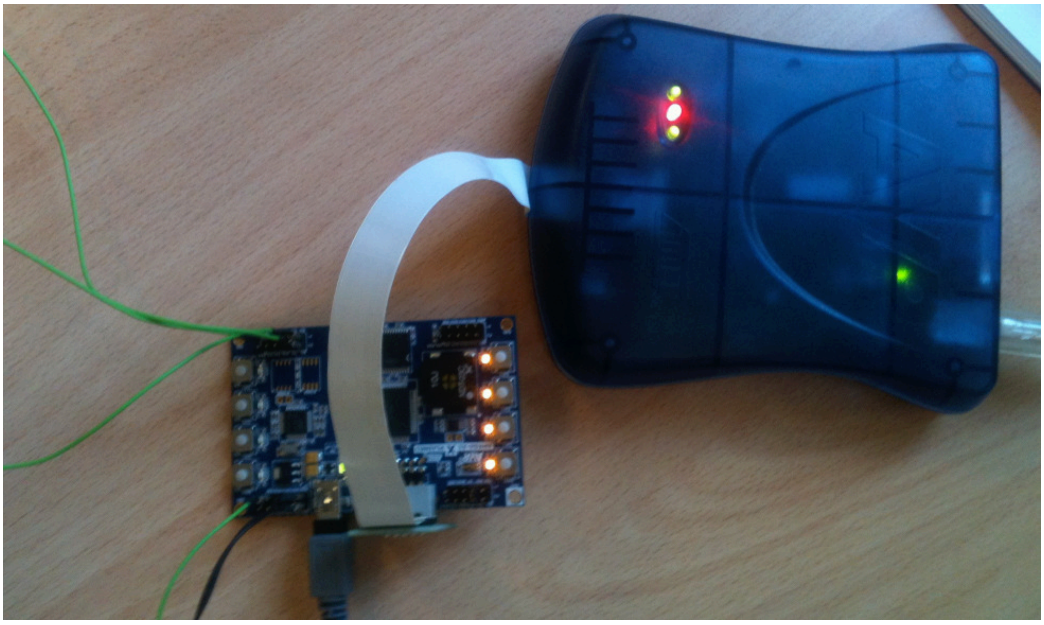


Abbildung 24: Atmel JTAGICE mkII Debugger und Xplainer Board

7 Fazit und Ausblick

Im abschließenden Kapitel werden die gewonnenen Erkenntnisse zusammengefasst und aus der Sicht des Autors bewertet. Anschließend wird ein Ausblick auf folgende Schritte und Tests, weitere Applikationen, wie auch weitere Entwicklungspotentiale des entworfenen Systems gegeben.

7.1 Fazit

Auch wenn das ASR noch nicht im Rennwagen verbaut ist und dort getestet werden konnte, so konnte mit den vorgenommenen Tests durchaus gezeigt werden, dass das Ergebnis dieser Arbeit ein funktionstüchtiges System ist. Alle Bauelemente der ASR wurden den Anforderungen gemäß ausgewählt und erfüllen diese auch. Das entworfene Programm ist in der Lage die anfallenden Programmschritte in Echtzeit abzuarbeiten. Das entworfene ASR entspricht somit der Zielstellung. Lediglich der Test des kompletten Systems im Rennwagen gilt noch vorgenommen zu werden.

7.2 Ausblick

Im Anschluss an diese Arbeit muss ein weiter Test zur Kompatibilitätsprüfung zwischen dem Xplained Baord und den Sensoren vorgenommen werden. Da die Tests im Vorfeld bereits zeigten, dass die einzelnen Systeme für sich funktionieren, wird dieser Test dies nur noch einmal verdeutlichen.

Schlussendlich muss das ASR in weiteren Testeinheiten im Rennwagen zeigen, ob es den Anforderungen unter realen Rennbedingungen standhält und diesen gerecht wird. Die Sensoren, die Signalgeberscheiben und der Microcontroller müssen dafür samt Verkabelung, Taster und Leuchtelementen am und im Rennwagen installiert werden. Für die Installation der Sensoren müssen im Vorfeld die Radträger vorbereitet werden. Anschließend soll das ASR unter realen Bedingungen getestet werden.

Um auf äußere Einflüsse reagieren zu können, soll eine Möglichkeit zur Verstellung des Eingreifens des ASR, also zur Veränderung des Schlupfgrenzwertes S, eingebunden werden. Dies könnte über ein Drehpotentiometer realisiert werden, welches über einen Analogeingang am Xplained Board angeschlossen werden kann.

Auch dient die ASR zur Vorbereitung weiterer elektronischer Hilfen. Auf dieser Basis soll später eine Automatikfunktion eingebunden werden. Zur Abarbeitung werden dann noch weitere Informationen wie Motordrehzahl und die Gangstellung benötigt.

Literatur

[FSAE12] Regelwerk der Formula Student 2012, http://www.fsaeonline.com/content/2012_FSAE_Rules_Version_90111K.pdf, Zugriff am 02.07.12

[MOTO12] http://www.motoport.de/ezine_bikes_test_0006.php4, Zugriff am 30.05.12

[TMM12] http://www.global.hs-mittweida.de/~tmm/tmm/joomla/index.php?option=com_content&view=article&id=257&Itemid=2, Zugriff am 03.07.12

[RWT08] Trzesniowski, Michael: Rennwagentechnik, Wiesbaden, Vieweg+Teubner, 2008, 1. Auflage, S. 721 f

[WASR12] http://de.wikipedia.org/wiki/Antriebsschlupfregelung, Zugriff am 28.06.12

[EZS05] Wörn, Heinz; Brinkschulte, Uwe: Echtzeitsysteme, Berlin Heidelberg, Springer- Verlag, 2005, S. 1f

[WNSA12] http://de.wikipedia.org/wiki/Nyquist-Shannon-Abtasttheorem, Zugriff am 11.07.12

[LM12] http://www.f2.htw-berlin.de/fileadmin/HTW/FB/FB2/Labore/Labor_Messtechnik/Sensor.pdf, Zugriff am 06.07.12

[LSW12] http://de.wikipedia.org/wiki/Lichtschranke, Zugriff am 10.07.12

[ATX09] Atmel, 8-bit AVR XMEGA A Microcontroller Handbuch, 2009

[ATD09] http://www.atmel.com/tools/xmega-a1xplained.aspx?tab=documents, Atmel ATxmega128A1 Xplained, Application notes und Datasheets, Zugriff am 27.08.2012

Anlagen

Anlage 1

Baumer

Passion for Sensors

Hochschule Mittweida (FH)
University of Applied Sciences
Technikum Mittweida
Herrn Robin Becker
Technikumplatz 17
09648 Mittweida

kmar-clg • direct +49 (0) 60 31/60 07-34 • kmartin@baumer.com 04.07.2012

Angebot

Sehr geehrter Herr Becker

Bezug nehmend auf Ihre E-Mail Anfrage vom 03.07.2012, bieten wir Ihnen freibleibend an:

Pos.	Produkt	Stück	Stückpreis
10	IFRM 04P15B1/L	4	68,70 € abzüglich 35 % Hochschulrabatt

Lieferzeit: ab Lager – Zwischenverkauf vorbehalten
Lieferung: ab Lager Friedberg plus Porto und Verpackung
Zahlung: 14 Tage 2 %, 30 Tage netto

Dieses Angebot ist 30 Tage gültig.

Ausführung des Auftrages und Lieferung erfolgen nur auf Basis unserer Allgemeinen Geschäftsbedingungen. Sollten Ihnen diese nicht vorliegen, so können sie bei uns kostenlos angefordert oder von unserer Internet-Seite heruntergeladen werden. Abweichungen hiervon bedürfen gesonderter schriftlicher Vereinbarung.

Bitte rufen Sie uns an, wenn Sie weitere Informationen benötigen. Wir freuen uns auf Ihren Auftrag.

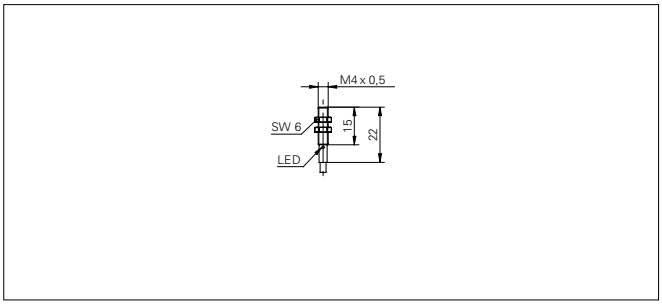
Mit freundlichen Grüßen
Baumer GmbH
i. A.
Claudia Günther
Vertriebsinnendienst Nord

Baumer GmbH • Pfingstweide 28 • D-61169 Friedberg
Phone +49 (0)60 31 60 07-0 • Fax +49 (0)6031 60 07-70 • sales.de@baumer.com • www.baumer.com
Deutsche Bank, Kto 776 1414, Blz 500 700 10 • SWIFT-BIC: DEUTDEFF • IBAN: DE 06500700100776141400
Geschäftsführer: Jörg Faber, Rüdiger Förster , Dr. Oliver Vietze • Amtsgericht Friedberg, Hessen HRB 1008 • Ust.-ID-Nr. DE 112 597 959

Anlage 2

Induktive Näherungsschalter IFRM 04P15B1/L

Masszeichnung



Allgemeine Daten	
Einbauart	bündig
Nennschaltabstand Sn	0,8 mm
Schalthysterese	2 ... 20 % von Sr
Schaltzustandsanzeige	LED rot

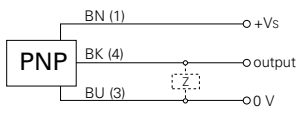
Elektrische Daten	
Schaltfrequenz	< 3 kHz
Betriebsspannungsbereich +Vs	10 ... 30 VDC
Stromaufnahme max. (ohne Last)	12 mA
Ausgangsschaltung	PNP Schliesser (NO)
Spannungsabfall Vd	< 2 VDC
Ausgangsstrom	< 100 mA
kurzschlussfest	ja
verpolungsfest	ja

Mechanische Daten	
Bauform	zylindrisch mit Gewinde
Material (aktive Fläche)	POM
Gehäusematerial	Chrom-Nickel-Stahl
Baugrösse	4 mm
Gehäuselänge	22 mm
Anschlussart	Kabel, 2 m

Umgebungsbedingungen	
Arbeitstemperatur	-25 ... +75 °C
Schutzart	IP 67

Foto

Anschlussbild



- Standard-Kabelausf. 22 mm

Anlage 3

Berechnung der maximalen Raddrehzahl ausgehend von der maximalen Geschwindigkeit

Max. Geschwindigkeit Rennwagen	v_{max} =	150	Km/h =	41,7	m/s
Durchmesser Rad	d_{rad} =			0,51	m
Radumfang u	u = Pi * d		u =	1,6	m
n_{max} =	$v_{max} = u * n_{max}$				
max Drehzahl Geberscheibe	>>> $n_{max} = v_{max} / u$ =			26,0	1/s
			n_{max} =	27	1/s
			nmax =	1620	1/min
max Raddrehfrequenz f_{max}	f = (z * n)/ 60				
Zähnezahl Geberscheibe z	bei z = 1 $f_{max} = n_{max}$ =			27	1/s = Hz
z = f bei einer Umdrehung / s	bei z = 24		f_{max} =	648	Hz
Clockspeed Microcontroller = Clsp			Clsp =	32	MHz

Rechenschritte pro Schaltflanke (RpS) = Clsp / (i * 2 * z * f *k)

i = Anzahl Raddrehzahlsensoren	i =	4	stk.
k= Korrekturfaktor (wg proportion Zahn zu Lücke)	k=	6	

z (Zähnezahl)	f_{max} [Hz]	RpI
1	27	24691
2	54	12346
4	108	6173
8	216	3086
12	324	2058
16	432	1543
20	540	1235
24	648	1029
32	864	772
48	1296	514

Berechnung der maximalen Raddrehzahl ausgehend von der maximalen Geschwindigkeit für das Eingreifen der ASR

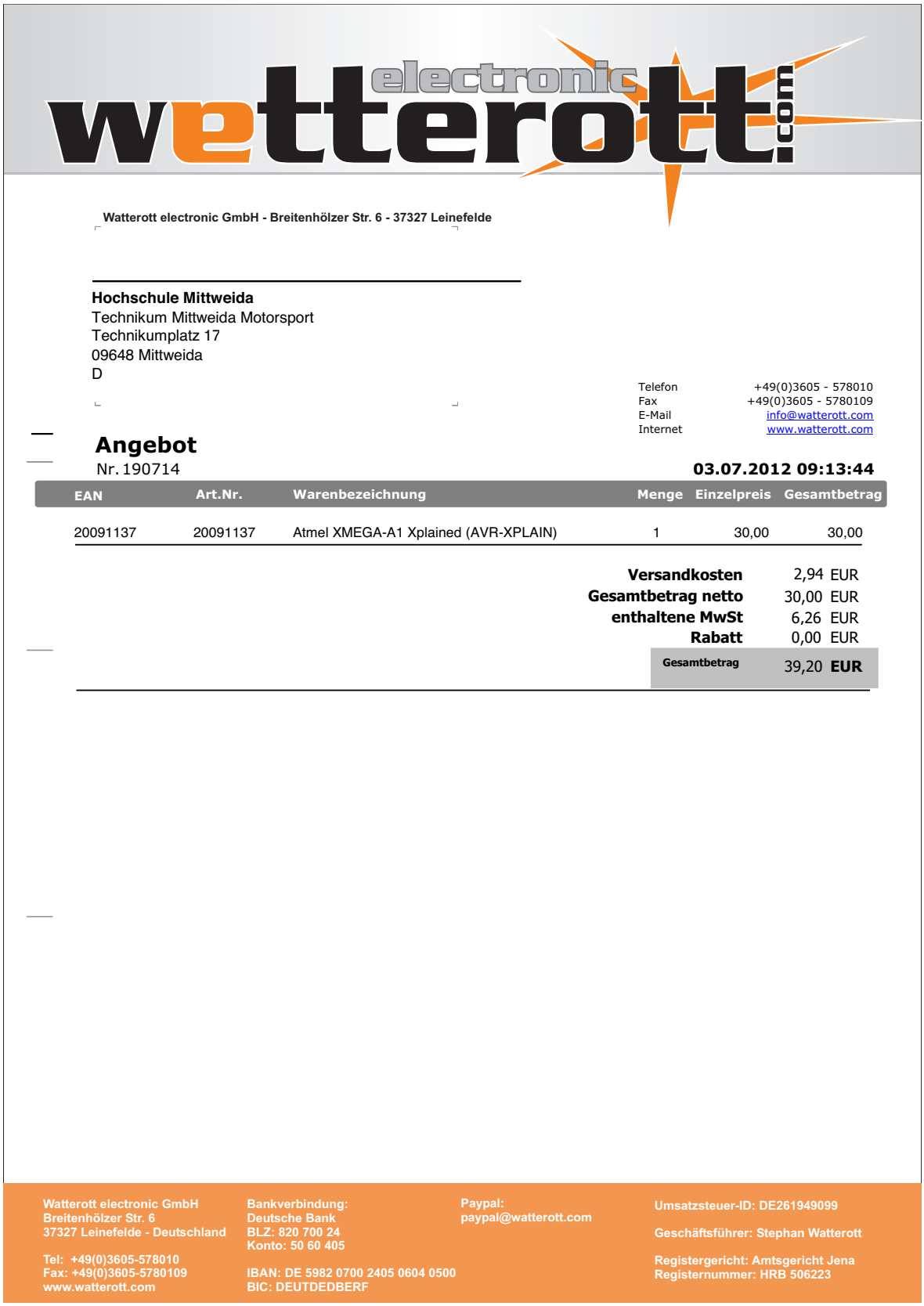
Max. Geschwindigkeit für ASR	v_{ASR} =		40 Km/h =	11,1	m/s
Durchmesser Rad	d_{rad} =			0,51	m
Radumfang u	u = Pi * d		u =	1,6	m
n_{ASR} =		$v_{ASR} = u * n_{max}$			
ASR Drehzahl Geberscheibe	>>>	$n_{ASR} = v_{ASR} / u$ =		6,9	1/s
			nasr =	7	1/s
			nasr =	420	1/min
max Raddrehfrequenz f_{ASR}		f = (z * n)/ 60			
Zähnezahl Geberscheibe z		bei z = 1 $f_{ASR} = n_{ASR}$ =		7	1/s = Hz
z = f bei einer Umdrehung / s		bei z = 24	f_{ASR} =	168	Hz
Clockspeed Microcontroller = Clsp			Clsp =	32	MHz

Rechenschritte pro Schaltflanke (RpS) = Clsp / (i * 2 * z * f *k)

i = Anzahl Raddrehzahlsensoren	i =	4	stk.
k= Korrekturfaktor (wg proportion Zahn zu Lücke)	k=	6	

z (Zähnezahl)	f_{ASR} [Hz]	RpI
1	7	95238
2	14	47619
4	28	23810
8	56	11905
12	84	7937
16	112	5952
20	140	4762
24	168	3968
32	224	2976
48	336	1984

Anlage 4



Watterott electronic GmbH - Breitenhölzer Str. 6 - 37327 Leinefelde

Hochschule Mittweida
Technikum Mittweida Motorsport
Technikumplatz 17
09648 Mittweida
D

Telefon +49(0)3605 - 578010
Fax +49(0)3605 - 5780109
E-Mail info@watterott.com
Internet www.watterott.com

Angebot

Nr. 190714 **03.07.2012 09:13:44**

EAN	Art.Nr.	Warenbezeichnung	Menge	Einzelpreis	Gesamtbetrag
20091137	20091137	Atmel XMEGA-A1 Xplained (AVR-XPLAIN)	1	30,00	30,00

Versandkosten	2,94	EUR
Gesamtbetrag netto	30,00	EUR
enthaltene MwSt	6,26	EUR
Rabatt	0,00	EUR
Gesamtbetrag	39,20	**EUR**

Watterott electronic GmbH
Breitenhölzer Str. 6
37327 Leinefelde - Deutschland

Tel: +49(0)3605-578010
Fax: +49(0)3605-5780109
www.watterott.com

Bankverbindung:
Deutsche Bank
BLZ: 820 700 24
Konto: 50 60 405

IBAN: DE 5982 0700 2405 0604 0500
BIC: DEUTDEDBERF

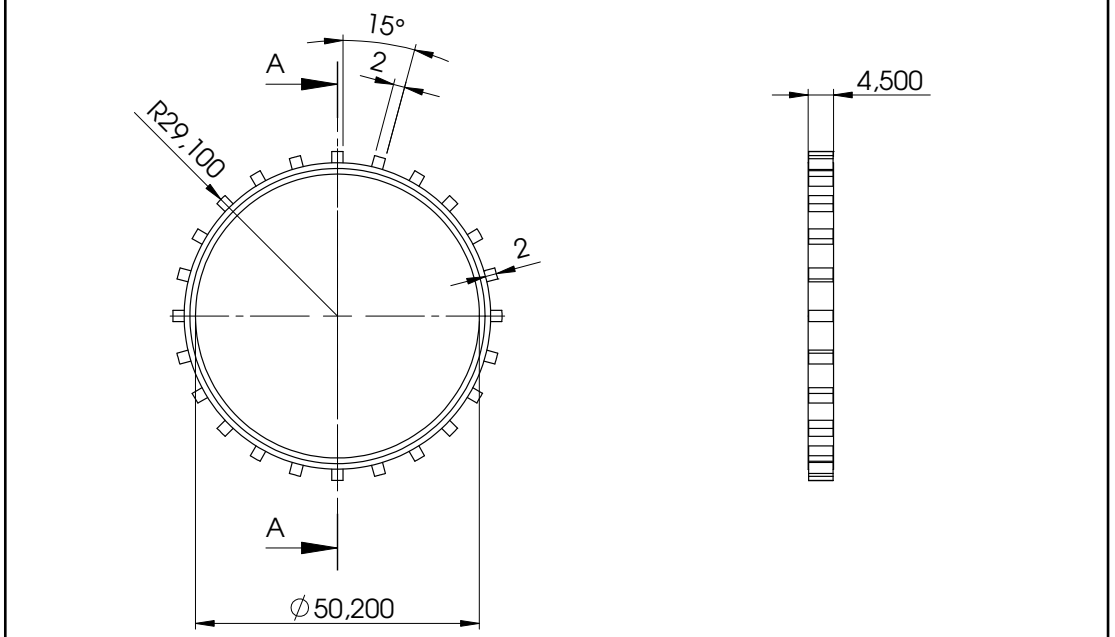
Paypal:
paypal@watterott.com

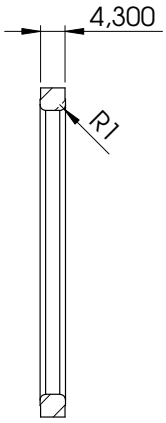
Umsatzsteuer-ID: DE261949099

Geschäftsführer: Stephan Watterott

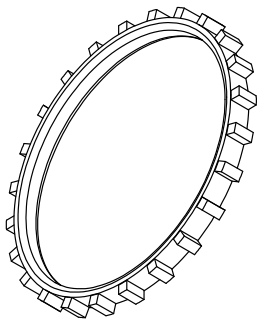
Registergericht: Amtsgericht Jena
Registernummer: HRB 506223

Anlage 5



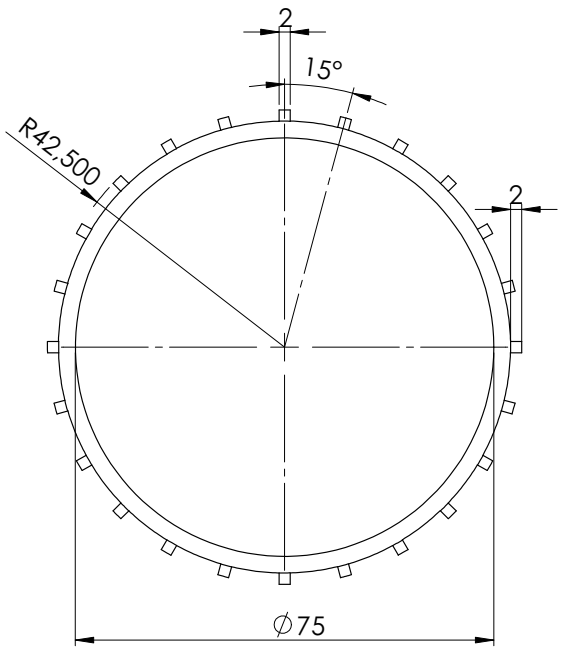


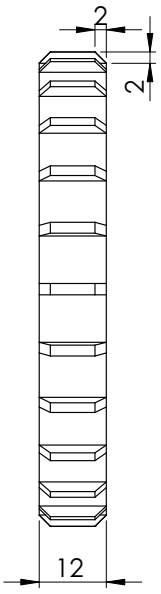
A-A

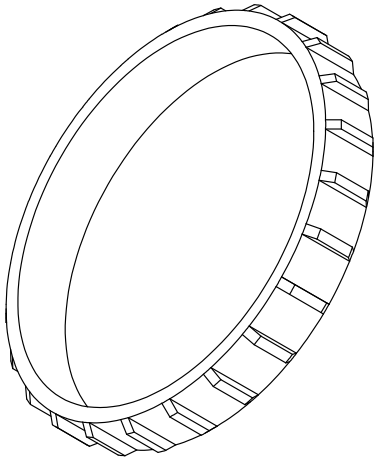


WENN NICHT ANDERS DEFINIERT: BEMASSUNGEN SIND IN MILLIMETER OBERFLÄCHENBESCHAFFENHEIT: TOLERANZEN: LINEAR: WINKEL:		OBERFLÄCHENGÜTE:			ENTGRATEN UND SCHARFE KANTEN BRECHEN	ZEICHNUNG NICHT SKALIEREN	ÄNDERUNG
	NAME	SIGNATUR	DATUM			BENENNUNG:	
GEZEICHNET						Distanzering innen klein-lager_vorn_ba_24.SLDPRT	
GEPRÜFT							
GENEHMIGT							
PRODUKTION							
QUALITÄT				WERKSTOFF:		ZEICHNUNGSNR.	A4
				EN AW 7075			
				GEWICHT:		MASSSTAB:1:1	BLATT 1 VON 1

Anlage 6







WENN NICHT ANDERS DEFINIERT: BEMASSUNGEN SIND IN MILLIMETER OBERFLÄCHENBESCHAFFENHEIT: TOLERANZEN: LINEAR: WINKEL:		OBERFLÄCHENGÜTE:		ENTGRATEN UND SCHARFE KANTEN BRECHEN		ZEICHNUNG NICHT SKALIEREN	ÄNDERUNG
	NAME	SIGNATUR	DATUM			BENENNUNG: Distanzring_hinten_ba_24	
GEZEICHNET							
GEPRÜFT							
GENEHMIGT							
PRODUKTION							
QUALITÄT				WERKSTOFF: EN AW 7075		ZEICHNUNGSNR.	A4
				GEWICHT:		MASSSTAB:1:1	BLATT 1 VON 1

Anlage 7

```
//************************************************************
// Project:     BA R.BECKER - ASR - ATXMEGA128A1
// Project file:   V1
// Source file:
// Editor:
// Date:
// Software: AVRGCC
// Hardware: ATXmega Unit
// Note:
//************************************************************
// notwendige Includes
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
//************************************************************
// Variablen und Definitionen

//************************************************************
// Funktionsprototypen

//************************************************************
// Hauptprgramm

#define LED  PORTE                      // LED -> PORTE
#define PINCONF 0b00010001              // Pin Config fuer alle Eingaenge
                                        // 0(SRLEN) 0(INVEN) 010(OPC) 001(ISC)
#define CLKSEL  0b00000110              // 000(),0(),0111(Taktquelle DIV256)
                                        // -> Prescaler: clk/256

void ConfigClockSystem( void );         // UP interne clock von 2MHz auf 32MHz

int main(void)
{

// Variablen Deklaration
   ConfigClockSystem();                 // UP Aufruf
   volatile uint32_t VL;                // Periodendauer Sensor vorn links
   volatile uint32_t VR;                // Periodendauer Sensor vorn rechts
   volatile uint32_t HL;                // Periodendauer Sensor hinten links
   volatile uint32_t HR;                // Periodendauer Sensor hinten rechts
   volatile uint32_t V;                 // Zwischenwert gemittelt Vorderachse
   volatile uint32_t H;                 // Zwischenwert gemittelt Hinterachse
   volatile int32_t X;                  // Prozentuale Abweichung von Hinter- zu Vorderachse
   volatile uint16_t S = 15;            // Schlufgrenzwert
   uint8_t i;                           // Zaehlwert 1
   uint8_t j;                           // Zaehlwert 2


// Definition Ausgaenge
   LED.OUT = 0b11111111;                // LED Ausgaenge vorbelegen
   LED.DIRSET = 0b11111111;             // LED als Ausgaenge setzen
   PORTF.OUT = 0b11111111;              // PORTF Ausgaenge vorbelegen
   PORTF.DIRSET = 0b11111111;           // PORTF als Ausgaenge setzen

// Definition Eingaenge
   PORTC.DIRSET = 0b00000000;           // PORTC als Eingaenge setzen
   PORTD.DIRSET = 0b00000000;           // PORTD als Eingaenge setzen

// Inputpins fuer Sensorignale configuration zuweisen
   PORTC.PIN0CTRL = PINCONF;
   PORTC.PIN1CTRL = PINCONF;
   PORTD.PIN0CTRL = PINCONF;
   PORTD.PIN1CTRL = PINCONF;



//T/C initialisieren
   //PORTxx als Eingang fuer event channel n setzen
   EVSYS.CH0MUX = 0b01100000;           // 0110(PORTC) 0() 000(PIN0)
   EVSYS.CH1MUX = 0b01100001;           // 0110(PORTC) 0() 001(PIN1)
   EVSYS.CH2MUX = 0b01101000;           // 0110(PORTD) 1() 000(PIN0)
   EVSYS.CH3MUX = 0b01101001;           // 0110(PORTD) 1() 001(PIN1)

//TCCx fuer "frequency capture" vorbelegen
   // 101(frequency capture) 0(EVDLY) 1XXX(EVSEL-Eventquelle Event CH0- CH7)
   TCC0.CTRLD = 0b10101000;
   TCC1.CTRLD = 0b10101001;
   TCD0.CTRLD = 0b10101010;
```

```
   TCD1.CTRLD = 0b10101011;

//Freigabe "capture or compare" Kanal X
   // 0001 (CCAEN - Kanal A), 0(),000 (WGMODE -> Normal)
   TCC0.CTRLB = 0b00010000;
   TCC1.CTRLB = 0b00010000;
   TCD0.CTRLB = 0b00010000;
   TCD1.CTRLB = 0b00010000;

// TOP Wert
   TCC0.PER = 0xFFFF;
   TCC1.PER = 0xFFFF;
   TCD0.PER = 0xFFFF;
   TCD1.PER = 0xFFFF;

//Taktquelle auswaehlen -> Timer starten
   // 000(),0(),0XXX(Taktquelle = CLKSEL)
   TCC0.CTRLA = CLKSEL;
   TCC1.CTRLA = CLKSEL;
   TCD0.CTRLA = CLKSEL;
   TCD1.CTRLA = CLKSEL;



// Start Dauerschleife

   do {
      do {
         // Sensorsignal zur ueberpruefung auf LEDs ausgeben
         LED.OUT = ~((PORTC.IN & 0b00000011) | (((PORTD.IN & 0b00000011) << 2) & 0b00001100));
         // Inputmuster zusammenfassen & auf Ausgang setzen
         // Eingaenge Maskieren, nur Ausgabe der Sensorsignale

      } while (((TCC0.INTFLAGS & TC0_OVFIF_bm)&&   // solange ein overflow Flag gesetz
           (TCC1.INTFLAGS & TC1_OVFIF_bm)&&
           (TCD0.INTFLAGS & TC0_OVFIF_bm)&&
           (TCD1.INTFLAGS & TC0_OVFIF_bm)));

      // Auslesen der Periodendauer
      VL = TCC0.CCA;
      VR = TCC1.CCA;
      HL = TCD0.CCA;
      HR = TCD1.CCA;

      // Berechnung für Vorder- und Hinterachse
      V =  ((VL +  VR) / 2);
      H =  ((HL +  HR) / 2);

      if ( H != 0 )                              // wenn H=0, Berechnung aussetzen
      {
         X = ((( V * 100) /  H  ) - 100);        // Berechnung der prozentualen Abweichung
         if (X > S)                              // wenn vorgegebende Schlupfgrenze überschritten
         {
            PORTF.OUT = 0b11111110;              // Ausgang schalten (Steuergeraet -> Zuendunterbrechung)
            LED.OUT = 0b10111111;                // LED Schalten

            // 1. Wartezeit (30*1s)
            for (i=0;i<30;i++)
            {
               _delay_ms(1);
            }

            PORTF.OUT = 0b11111111;              // Ausgang und LED ruecksetzen
            LED.OUT = 0b11111111;

            // 2. Wartezeit (250*1s)
            // wegen Zuendunterbrechungsvorbelegung auf Steuergeraet
            for (j=0;j<250;j++)
            {
               _delay_ms(1);
            }
         }
      }
   } while (1);
}  // end main
```

```
void ConfigClockSystem( void )               //entnommen aus "XMEGA Timer/Counter extension driver example
source"
{
   // Start internaue 32MHz RC oscillator.
   OSC.CTRL = OSC_RC32MEN_bm;

   do {
   // Wait while oscillator stabilizes.
   } while ( ( OSC.STATUS & OSC_RC32MRDY_bm ) == 0 );

   // Enable prescaler B and C.
   CCP = CCP_IOREG_gc;
   CLK.PSCTRL = CLK_PSBCDIV_2_2_gc;

   // Select 32 MHz as master clock.
   CCP = CCP_IOREG_gc;
   CLK.CTRL = CLK_SCLKSEL_RC32M_gc;

}

//*************************************************************
// Funktionen

//*************************************************************
// EOF
```

Danksagung

An dieser Stelle möchte ich zuerst meiner Familie danken, da sie mich stets unterstützte und ohne welche dieses Studium für mich kaum möglich gewesen wäre.

Für das Korrekturlesen möchte ich mich bei Henriette Hahn und Sebastian Kipping bedanken, da sie mir damit beide einen großen Freundschaftsdienst erwiesen haben.

Auch dem gesamten Team von TMM gebührt Dank, denn ohne meine persönliche Entwicklung, welche ich während meiner zwei jährigen Tätigkeit im Team erfahren durfte, wäre der Grundstein für diese Arbeit wohl nicht gelegt wurden. Stets wurde auch mit gutem Rat und Unterstützung zur Seite gestanden.

Besonderer Dank gilt an dieser Stelle auch allen Angestellten und Mitarbeitern des Application Center Microcontroller (ACMC), welche mich bei der Programmierung des Microcontrollers unterstütz und mir die nötige Technik zum Test dessen zur Verfügung gestellt haben.

Abschließend gilt nur zu sagen, dass es wie meistens im Leben stets anders kommt als man denkt. In diesen Momenten ist es schön zu wissen, dass es Leute gibt die einem zur Seite stehen.

Eidesstattliche Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Literatur und Hilfsmittel angefertigt habe.

Stellen, die wörtlich oder sinngemäß aus Quellen entnommen wurden, sind als solche kenntlich gemacht.

Diese Arbeit wurde in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt.

Mittweida, den 12.09.2012

Robin Becker